Systems and Methods of Interaction, Report + Demonstrators

Authors: Raphael Höver, Manuel Ferre, Roque Saltarén, Salvador Cobos, and Hector Moreno, Benjamin Bayart, Jean-Rémy Chardonnet, Adrien Escande, Abderrahmane Kheddar

Date: M18

Due Date: M18

Del. /Task Identifier: D4.2.1

Work Package: WP4

Partner(s): ETH, LSC, UPM

Work Package Leader: ETH

Confidentiality Level: PU

Abstract:

According to task 4.3 an algorithm has been developed that generates an appropriate haptic feedback based on force signals that have been recorded along arbitrary trajectories in a real environment. The focus of this work is, to provide methods to generate haptic feedback which is purely based on previously recorded quantities. No models are employed for the haptic properties of the scene content.
According to task 4.4 the advances in the development of a multi-body dynamic hand model are described. A demonstrator of object manipulation has been developed and a realistic human hand model has been defined. The human hand model has been developed in order to realistically characterize and model hand postures during object grasping.
Concepts for haptic modulation are presented as proposed in task 4.5. A terminology is established which distinguishes between the pure manipulation of haptic signals by data processing techniques and the augmentation of the raw haptic feedback by signals based on additional information. Formulas are proposed how to merge two haptic signals from two different sources into one signal which is provided to the user. In addition, some applications for these concepts are given that can be used to conduct according experiments.
Complying with task 4.6 a method is described that integrates haptic cues into contact space

formulations to drive the dynamic interaction of avatars within an environment with different materials. A constraint-based method is chosen to solve contact situations. All the avatar dynamics and the haptic cues are integrated into one contact space formulation while a high performance of the underlying algorithms is maintained.

This page is left blank intentionally.

Table of Contents

1 Executive Summary

According to the rolling workplan for months 13-30 the following tasks are covered in this deliverable

- T4.3 Methods for haptic replay
- T4.4 Software tools for haptic interactions based on multi-body dynamics models
- T4.5 Theory and algorithms for modulated haptics and haptic augmentation in mixed reality multimodal presence environments
- T4.6 Haptic enhanced avatars

The goal of task 4.3 is to study methods of generating unobserved haptic feedback based on recorded data sets. The recorded data is decomposed into interaction patterns and a relationship between these interaction patterns and the corresponding measured feedback is established. Interpolation methods are analyzed with regard to requirements in the haptic record-create chain to extend the scattered data of the recorded signals into a continuous domain.

Task 4.4 aims to continue the previous work on human hand modelling and finalize the first software tools based on multi-body dynamics. As continuation of the previous work, a hand gesture procedure is developed in order to optimize the haptic interaction. The aim of this study is to reflect forces according to hand gesture while objects are being manipulated. Further study is, however, necessary in order to improve and simplify hand gesture identification.

Fundamental research on haptic augmentation is conducted in task 4.5 using hardware developed in WP3. A basic setup for augmented reality haptics is being developed, which allows the study of key elements in such a setup. A theory for haptic modulation and augmented reality haptics has been formulated: The haptic modality is not only used as a component but also as a mean to bring extra-haptic cues that can help the user to complete a given task. First technical algorithms for blending real and virtual haptic cues are proposed.

Task 4.6 focuses on modeling contact space formulation to be used to drive the dynamic interaction between bodies of different kinematics structure (focusing on virtual avatars) with different materials composing the surrounding environment. The contact space formulation needs to subtly integrate the avatar dynamics together with other phenomena such as impacts, static and dynamic friction, deformations. An autonomous behaviour of the avatar is developed, such as touching the environment to engender motion in an autonomous way. An implementation of a contact support planner (CSP) has been done. Given an initial posture and a final goal, the CSP will find for a possible trajectory where the avatar uses its environment to move toward the goal, by choosing appropriate support contact points.

2 Introduction

In section 3 a basic framework is presented that generates an appropriate haptic feedback based on force signals that have been recorded along arbitrary trajectories in a real environment. The focus of this work is, to provide methods to generate haptic feedback which is purely based on previously recorded quantities. No models are employed that describe the haptic properties of the scene content. This guarantees a framework which is not dependent on any assumptions about the objects that the user interacts with. Besides an algorithm to display quasi static feedback to the user an extension of this algorithm is presented which allows incorporating slip during the replay session. Several aspects are mentioned how to enhance the current state of the framework to improve the quality of the feedback yet further.

Sections 4 and 5 deal with the advances in the development of a multi-body dynamic hand model. Work that has been carried out by the UPM in this task is complementary. First, a demonstrator of object manipulation has been developed, and second a realistic human hand model has been defined. Object manipulation demonstrator is based on a multi-body dynamics software library which describes the interaction between a haptic device and a virtual manipulated part. The human hand model has been developed in order to realistically characterize and model hand postures during object grasping.

Concepts for haptic modulation are presented in section 6. A terminology is established which distinguish between the pure manipulation of haptic signals by data processing techniques and the augmentation of the raw haptic feedback by signals based on additional information which is provided. Furthermore, formulas are proposed how to merge two haptic signals from two different sources into one signal which is provided to the user. Examples for both strategies are presented and applications for an implemented demonstrator covering all three scenario classes P2O, P2P and POP are discussed.

In section 7 a technique is described that integrates haptic cues like friction into contact space formulations to drive the dynamic interaction of avatars within an environment with different material properties. A constraint-based method is chosen to solve contact problems. All the avatar dynamics and the haptic cues are integrated into one contact space formulation while a high performance of the underlying algorithms is maintained. The algorithm is capable to accurately compute the contact forces of the avatar with its environment including friction.

3 Methods for haptic replay

In the last eighteen months an algorithmic framework has been developed that generates an appropriate haptic feedback based on force signals that have been recorded along arbitrary trajectories in a real environment. The framework does not incorporate any parameterized models for the haptic rendering but works directly on previously captured signals. This method allows the development of the most flexible algorithms which are not constrained by any scene abstractions and kinetic primitives like springs or dampers. Also, the data representation of the object geometry given by the visual capturing system is completely independent from the haptic rendering system since the haptic system does not incorporate mass spring meshes or any other structures that are constrained by the object geometry. The interface of the object geometry to the haptic rendering algorithm is an algorithm for collision detection. A collision point has to be determined to span a local interaction space and look up the correct set of recorded haptic stimuli to compute an appropriate feedback. Texture coordinates on the object surface enable to generate haptic feedback which is also dependant on the current contact position. The main components of the algorithmic framework for the haptic replay sessions are shown in Figure 3.1.

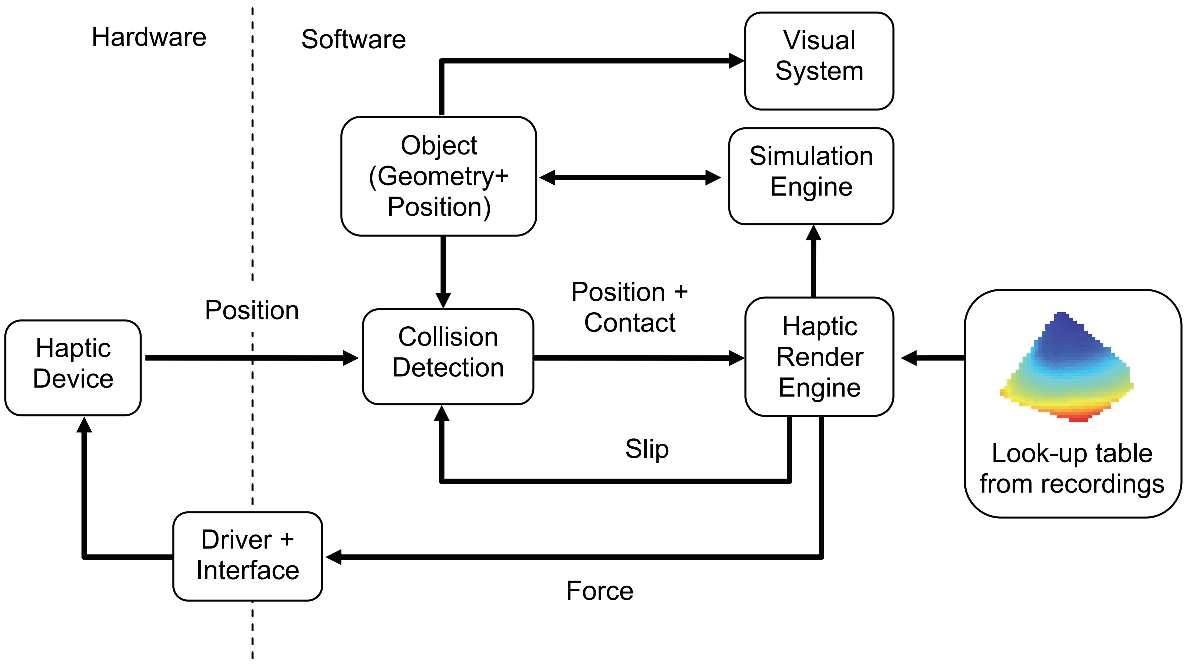


Figure 3.1 Algorithmic framework for haptic replay

The design of the framework as shown in Figure 3.1 is based on [1]. During a replay session the position of the interaction point is tracked by the haptic device and evaluated by the collision detection algorithm. If no collision occurs no feedback is generated. If the interaction point collides with the object geometry a contact point is calculated which serves as the origin of a local surface coordinate system. The contact point and the position of the interaction point in this local coordinate frame are transfered to the haptic render engine. This module determines the haptic feedback by interpolating a corresponding force vector from the previously recorded force signals within this local domain. Since the contact point is provided as well, this feedback does generally not only depend on the current interaction (position, velocity…) of the user but also on the current position on the object surface. During the calculation of the haptic feedback the haptic render engine tests whether the contact point for the current interaction is valid. As soon as the contact point becomes invalid with respect to

the current interaction the render engine switches back to the collision detector which then handles slip along the object surface or breaks the contact. If the contact is valid the calculated force vector is transferred to the driver of the haptic device where it is transformed into the torque space of the device and finally rendered to the user.
After this general description of the framework for haptic replay the following sections will describe some modules of Figure 3.1 more detailed.

3.1 Collision Detection

To assess if any feedback should be calculated the first step of the replay algorithm is a collision detection. To perform this detection a representation of the object geometry and the position of the object is necessary. The information about the geometry will be provided by the visual recording system, which generates a triangle mesh of the object during the recording session. The position (and orientation) of the object is currently assumed to be static and thus does not need to be estimated by a simulation engine. This will be part of future implementation.
The algorithm for the collision detection looks for an intersection of the line segment between the current and the last position of the interaction point with the object geometry. This collision test is skipped as long as a valid contact is available. If the haptic render engine gets an invalid interaction for the current contact point it switches back to the collision detection algorithm which iteratively moves the contact point across the object surface until either a valid haptic feedback can be calculated or the contact breaks up.

3.2 Haptic Render Engine

The haptic render engine is the core module of the developed haptic replay framework. In contrast to common algorithms that are used for calculating haptic feedback the approach taken in this project does not incorporate any models for different haptic cues like stiffness or slip. Thus, no parameters for such models like spring constants or friction coefficients need to be estimated. Instead, previously captured signals are directly used to interpolate the haptic feedback that corresponds to the interaction of the user. An abstract interaction space needs to be defined where each interaction pattern of the user is represented by a vector. Components of this vector can represent the current position or velocity but also any other linear combination of the measured position signal. During the record session the movement of the interaction point is captured and for each time step the corresponding interaction vector is calculated. The measured force vector that belongs to the interaction vector is stored in the interaction space at the corresponding position. Hence, the primary data structure of the recorded values is a scattered vector field in the interaction space as seen in figure 2.2. It is important to mention that so far no slip is allowed during the recording session. As soon as the user gets into contact with the real object the contact point is assumed to remain constant until the contact breaks.
During the replay the user is not constrained to only execute previously recorded interaction patterns. Thus it is necessary to interpolate the recorded force vectors in the interaction space. For this necessary interpolation different techniques have been examined. Suitable methods for the interpolation of scattered data in an n-dimensional domain are

- Inverse Distance Weights
- Moving Least Squares
- Natural Neighbour Methods
- Radial Basis Functions
- Simplex-Based Interpolation

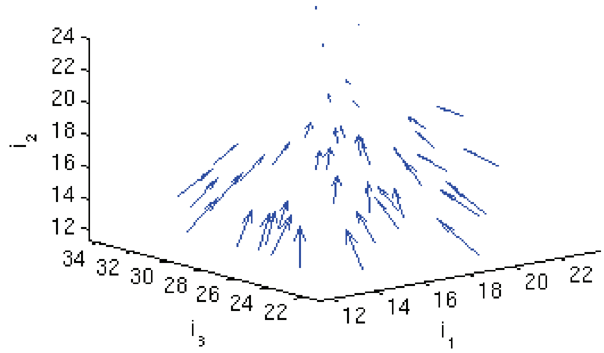


Figure 3.2 Typical recorded data set. The arrows represent the force vectors recorded at the corresponding position in the interaction space.

Until now, only simplex based methods have been considered due to their rather simple implementation and their efficient computation for large data sets. This class of interpolation techniques is capable to interpolate the scattered data inside the convex hull of the data set. Outside of the convex hull no valid data is supplied. Thus, if the user generates interaction patterns that lie outside of the convex hull of the recorded data set the generated feedback of the system has to be determined by an additional rule. Possible techniques are nearest neighbor methods to extrapolate the recorded data set into the whole interaction space. However, for dimensions of the interaction space that reflect the current position of the interaction point, it is possible to exploit this restriction to a convex subspace to incorporate slip during the replay session. Assuming the user explored the valid interaction space to its whole extent without creating any slip or damage during the recording the convex hull of the captured data set represents the subspace wherein static friction and no damage of the object occurs. Of course, if the real friction hull is concave instead of convex this algorithm will not give the correct hull.

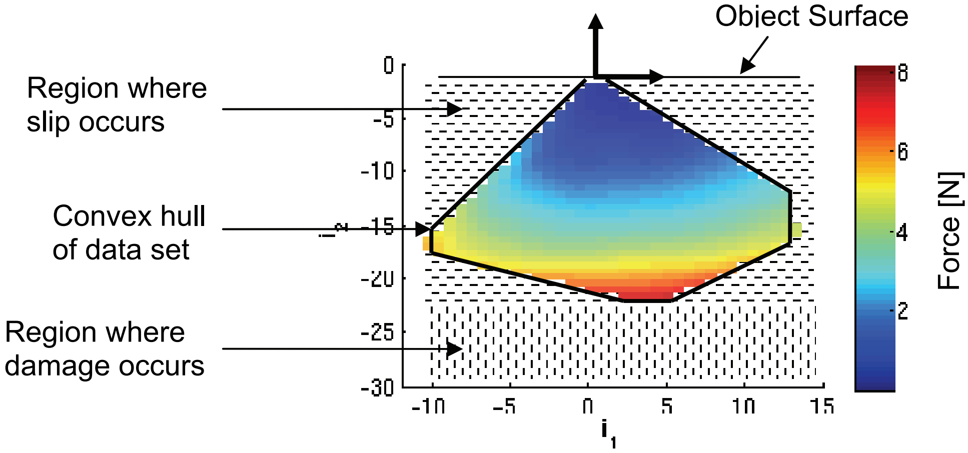


Figure 3.3 Interpolated convex data set. The interaction dimensions i_1 and i_2 describe the position of the interaction point relative to a local coordinate system on the surface. The magnitudes of the interpolated force vectors are color coded. Outside of the convex subspace either slip or damage occurs.

Thus, if an interaction outside of this convex subspace occurs during the replay the system should handle this event either as slip across the object surface or as damage of the object. The inherent concept of this strategy to render slip and damage is similar to the algorithms presented in [1] and [3] but no parameters have to be estimated and the region for static friction is not limited to any specific shapes like cones (which result from classical friction models with a constant friction parameter μ and a linear material stiffness in normal direction).

To handle damage of the material adequately more measurements would be necessary to observe the actual force signal during cutting through the material. At this stage of the replay framework no effort is planed to add a suitable extension to the algorithm. Thus, subsequently only the case of slip is discussed.
In the following it is assumed that the current interaction point lies outside of the interpolated convex region and according to Figure 3.3 inside the region where slip occurs. In this case the interpolated region is shifted laterally so the current interaction point lies on the convex hull of the interpolated region. This shift is illustrated in Figure 3.4.

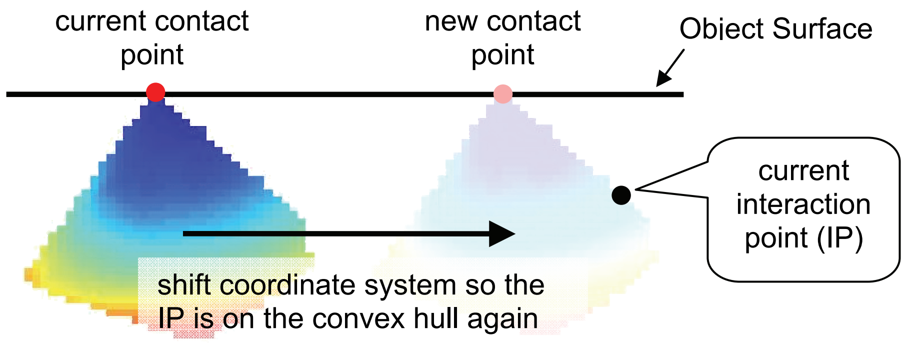


Figure 3.4 Generation of slip: If the current interaction point (IP) is not inside the convex subspace of interpolated force vectors the contact point and thus the subspace is shifted so the IP lies on the convex hull of the data set.

All the described techniques have been implemented in a demonstrator system. However, it is still necessary to evaluate the performance of the described techniques by psychophysical studies. Corresponding experiments are currently designed and will be conducted soon.

3.3 Driver and Interface

To assure a reliable and stable haptic feedback during the replay session with low jitter in the acquired data samples a real time system with according hardware drivers has been implemented. The data acquisition of all sensors needs to be synchronously driven by a single hardware clock and the sample rate should be adjustable to the needs of the considered scenario. Thus, the standard drivers of the considered Phantom interfaces are not used but generic hardware drivers for data acquisition cards are employed in the system.
Also, the supplied servo amplifiers that are part of the Phantom device are not suitable for the desired high fidelity haptic feedback. Since these amplifiers control the voltage at the motor clamps instead of the current the rendered force is disturbed if the end effector of the device is moved in the workspace. A detailed explanation of the resulting perturbation of the force signal is given in [4]. According to this, new servo amplifiers are currently developed to provide an accurate feedback.

3.4 Future Implementations

The current implementation of the haptic replay algorithm is limited to static scenes. The simulation engine shown in Figure 3.1 has not been implemented yet. However, regarding the scenarios described in deliverable 6.2.1 for P2O it is mandatory to handle object dynamics during the replay as well. Otherwise, tasks like lifting and moving objects could not be performed in according experiments. Following the concept to render all the feedback purely data driven the object dynamics would have to be generated based on the observed recordings and without any dynamic models. However, this would require a vast amount of

captured observations to reliably predict the object dynamics that result from the acting forces. To enable early experiments according to D6.2.1 a preliminary physical simulation engine will be implemented in this phase of the project.
Furthermore, it is questionable if the described method for rendering slip across the object's surface provides highly realistic feedback. The current concept neglects any surface texture that would modulate the forces that occur if the user moves across the surface. Hence, it is planned to extend the current haptic render engine with multiple feedback tracks so recorded force profiles from slipping situations can be overlaid on the quasi-static feedback that is rendered by the current algorithm.

3.5 Demonstrator

The algorithms and software components described in sections 3.1 to 3.3 have been implemented into a demonstrator system for P2O scenarios. As the haptic interface a Phantom Desktop 1.5 is employed thus only tool based interactions with one interaction point are addressed so far. The device has six degrees of freedom for the position and orientation sensing and three degrees of freedom for the force actuation. The haptic device is connected to an RTAI Linux real time system via a generic data acquisition card from Sensoray (Model 626) which reads the encoder values for determining the current tool position. Also, this card is connected to the three servo amplifiers of the device to control the rendered forces. To access the Sensoray card the Comedi driver library is used since these drivers are capable to run under a real time system. While the pure data acquisition and force rendering algorithms run on the real time system a second machine hosts the main algorithm for calculating collisions and slip and interpolating the correct force vectors for the current feedback. This second computer is connected as a client to the real time machine and receives the sampled position signal from the device and sends back the corresponding force as illustrated in Figure 3.5.

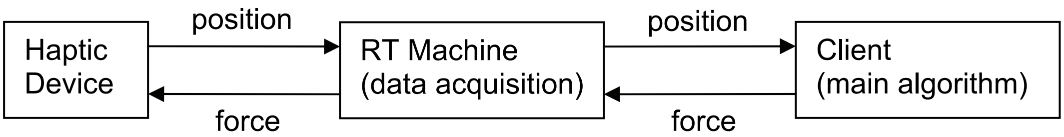


Figure 3.5 Design of P2O demonstrator system for haptic replay

The system runs easily with sample frequencies up to 1000 Hz. Technically, even higher sample frequencies are possible to render stiffer materials (according to [5]) but experiments reveal that the mechanical properties of the Phantom arm are not suitable for such high update rates with large rendered stiffnesses. The arm tends to vibrate and a reasonable feedback is not possible.
Since the haptic system has not been merged with the visual system yet, synthetically generated triangle meshes that represent the object's geometry are used to enable the collision detection during the replay session.

4 Software tools based on multi-body dynamics

Multi-body dynamic model to control realistic, full hand manipulation of rigid objects (Demonstrator)

A demonstrator has been developed in order to analyze manipulation of rigid objects using haptic devices. This demonstrator is made up of a haptic interface held by a user who is interacting with a virtual environment that simulates the object manipulation.

At the moment, a first version of libraries for simulating the dynamics of the manipulation has been developed. These libraries represent the HandModelMSIM block shown in Figure 4.1. Next figure shows a block diagram of the demonstrator structure. This demonstrator is based on a haptic interface that interacts with a virtual environment. The interface provides the corresponding forces and positions in order to properly interact with the virtual object. User is in charge of deciding the force movement actuation according to manipulation object.

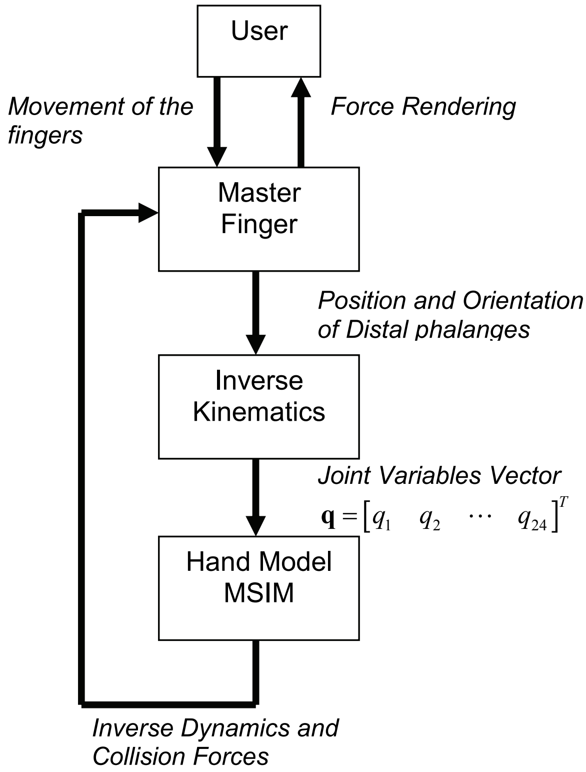


Figure 4.1 General scheme for object interaction demonstrator.

4.1 Modelling of a human hand based on multibody dynamics equations

A multi-body system is an assembly of two or more rigid bodies (also called elements) connected by kinematic joints, having the possibility of relative movement between them. A joint permits certain degrees of freedom of relative motion and prevents or restricts others. The degrees of freedom denote the number of independent kinematical possibilities to move. A rigid body has six degrees of freedom in the case of general spatial motion, three of them translational degrees of freedom and three rotational degrees of freedom. In the case of planar motion, a body has only three degrees of freedom with only one rotational and two translational degrees of freedom. When a multibody dynamic system is intended to perform a desired task, the source of motion is provided by actuators. This actuator can be rotational or prismatic and are mounted on the joints of the system.

Joints and actuators imply constraints of motion in the kinematical degrees of freedom of one or more bodies. The classical constraint is usually an algebraic equation that defines the relative translation or rotation between two bodies. When a velocity constraint can be integrated in time in order to form a position constraint, it is called holonomic constraints, otherwise non-holonomic.

An example of multibody system can be a hand where the phalanges can be considered as the bodies and the articulations as kinematic joints. In Figure 4.2, a hand of 24 degrees is showed. This model is based on the previous work define in this WP in month 12, it has been published in [11].

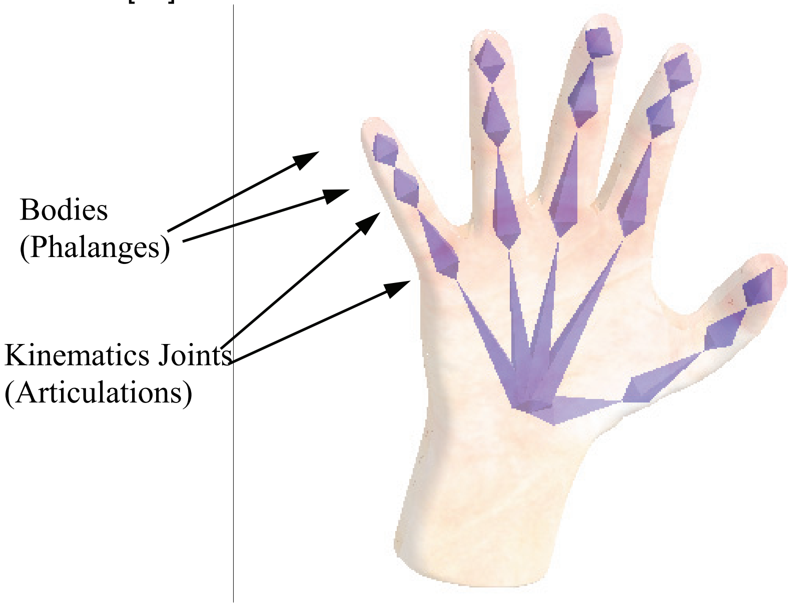


Figure 4.2 A hand considered as a multi-body system.

One the main subjects in multi-body dynamics is to determine system's posture (position an orientation of all the bodies) during certain amount of time. To describe the posture of a body a reference frame is attached to the body. Therefore the position of a body is represented by a position vector of the body's frame origin. On the other hand, the orientation can be described in several forms such as the axis-angle representation, the Euler angles, and in terms of the Euler parameters. In this work the orientation is given in terms of the Euler parameters.

The Euler parameters are based on the Euler theorem which states that any orientation of a body can be achieved by a single rotation from a reference orientation (expressed by an angle χ) about some axis (defined by a unit vector $\mathbf{u}$.) The Euler parameters are given in the following form:

$$\mathbf{p} = \begin{bmatrix} e_0 & \mathbf{e}^T \end{bmatrix}^T,$$

where $e_0 = \cos\left(\frac{\chi}{2}\right)$ and $\mathbf{e} = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix} = \mathbf{u}\sin\left(\frac{\chi}{2}\right)$.

4.2 Newton-Euler form of motion constrained equations in terms of Euler parameters

The Newton-Euler equation of motion permits to model the dynamic behaviour a multi-body system [14]. This equation models the relationship between a multi-body system movement (i.e., positions, velocities and acceleration of all the bodies during a certain quantity of time) and the external forces applied on the system. This equation regards the inertial proprieties of the bodies and the way they are connected.

Before showing the equation of motion, consider a multi-body system of *nb* bodies, the composite set of generalized coordinates is

$$\mathbf{r} = \begin{bmatrix} \mathbf{r}_1^T & \mathbf{r}_2^T & \cdots & \mathbf{r}_{nb}^T \end{bmatrix}^T \tag{4.1}$$

$$\mathbf{p} = \begin{bmatrix} \mathbf{p}_1^T & \mathbf{p}_2^T & \cdots & \mathbf{p}_{nb}^T \end{bmatrix}^T \tag{4.2}$$

where the vector $\mathbf{r}_i$ and quaternion $\mathbf{p}_i$ represents the position and orientation of the frame attached to the body i. The kinematic and driving constraint that act on the system are given in the form

$$\mathbf{\Phi}(\mathbf{r}, \mathbf{p}, t) = \mathbf{0} \tag{4.3}$$

In addition, the Euler parameter normalization constraints must be hold

$$\mathbf{\Phi}^{\mathbf{p}} \equiv \begin{bmatrix} \mathbf{p}_1^T \mathbf{p}_1^T - 1 \\ \vdots \\ \mathbf{p}_{nb}^T \mathbf{p}_{nb}^T - 1 \end{bmatrix} = \mathbf{0} \tag{4.4}$$

regarding the aforementioned relationships the Newton-Euler form of constrained equations of motion in term of the Euler parameters are:

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{\Phi}_{\mathbf{r}}^T & \mathbf{0} \\ \mathbf{0} & 4\mathbf{G}^T\mathbf{J}'\mathbf{G} & \mathbf{\Phi}_{\mathbf{p}}^T & \mathbf{\Phi}_{\mathbf{p}}^{\mathbf{p}T} \\ \mathbf{\Phi}_{\mathbf{r}} & \mathbf{\Phi}_{\mathbf{p}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}_{\mathbf{p}}^{\mathbf{p}} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{r}} \\ \ddot{\mathbf{p}} \\ \boldsymbol{\lambda} \\ \boldsymbol{\lambda}^{\mathbf{p}} \end{bmatrix} = \begin{bmatrix} \mathbf{F}^A \\ 2\mathbf{G}^T\mathbf{n}'^A + 8\dot{\mathbf{G}}^T\mathbf{J}'\dot{\mathbf{G}}\mathbf{p} \\ \boldsymbol{\gamma} \\ \boldsymbol{\gamma}^{\mathbf{p}} \end{bmatrix} \tag{4.5}$$

Where:

$\mathbf{M} \equiv \operatorname{diag}(m_1\mathbf{I}_3, m_2\mathbf{I}_3, \ldots, m_{nb}\mathbf{I}_3)$, is the mass matrix, it is a composite set of mass matrix of the nb bodies of the system. The term m_i refers to the mass of the body i and $\mathbf{I}_3$ is a 3x3 identity matrix.

$\mathbf{J}' \equiv \operatorname{diag}(\mathbf{J}_1', \mathbf{J}_2', \ldots, \mathbf{J}_{nb}')$, is a composite set of the Inertia matrices of the nb bodies.

$\mathbf{F} \equiv \begin{bmatrix} \mathbf{F}_1^T & \mathbf{F}_1^T & \cdots & \mathbf{F}_{nb}^T \end{bmatrix}^T$, is the vector of external forces applied on the bodies

$\mathbf{n}' \equiv \begin{bmatrix} \mathbf{n}_1'^T & \mathbf{n}_2'^T & \cdots & \mathbf{n}_{nb}'^T \end{bmatrix}^T$, is the vector of external torque applied on the bodies

$\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}^{\mathbf{p}}$ are the Lagrange multipliers vector

$\boldsymbol{\gamma}$ and $\boldsymbol{\gamma}^{\mathbf{p}}$ are the acceleration vector

$\mathbf{G} \equiv \operatorname{diag}(\mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_{nb})$, is the composite of $\mathbf{G}_i = [-\mathbf{e} \quad -\mathbf{e} + e_o\mathbf{I}_3]$

$\mathbf{\Phi}_{\mathbf{r}}$ and $\mathbf{\Phi}_{\mathbf{p}}$ are the Jacobian matrices of $\mathbf{\Phi}$ whit respect to the position vector $\mathbf{r}$ and Euler parameters quaternion $\mathbf{p}$.

$\mathbf{\Phi}_{\mathbf{p}}^{\mathbf{p}}$ y the Jacobian matrix of $\mathbf{\Phi}^{\mathbf{p}}$ whit respect to quaternion $\mathbf{p}$.

This system of equations, taken with the kinematic and Euler parameter normalization constraints of Eq. 4.1and 4.2 and the associated velocity equations

$$\mathbf{\Phi}_{\mathbf{r}}\dot{\mathbf{r}} + \mathbf{\Phi}_{\mathbf{p}}\dot{\mathbf{p}} = -\mathbf{\Phi}_t = \nu \tag{4.6}$$

and

$$\mathbf{\Phi}_{\mathbf{p}}^{\mathbf{p}}\dot{\mathbf{p}} = \mathbf{0} \tag{4.7}$$

describes mixed algebraic motion equations of the Euler parameters terms.

Initial condition must be given on $\mathbf{r}$ and $\mathbf{p}$ at t_0 so that the equation 4.1 is satisfied. For velocities, initial conditions should be given on $\dot{\mathbf{r}}$ and $\dot{\mathbf{p}}$:

$$\mathbf{B}_{\mathbf{r}}^{I}\dot{\mathbf{r}} + 2\mathbf{B}_{\boldsymbol{\omega}'}\mathbf{G}\dot{\mathbf{p}} = \mathbf{v}^{I} \qquad (4.8)$$

With the equation of motion, Ec. 4.5, it was possible to solve the inverse and forward dynamics problem of a hand of 24 degrees of freedom. The inverse dynamic problem permits to compute the required forces in the actuators to generate a desired motion. On the other hand the forward dynamic problem computes the motion of the system produced by the force and torque in the actuators.

Figure 4.3 shows some examples of the manipulation simulation. Tasks are described according to the proper multi body equations. These equations have to take into account the movement constraints for each configuration. As results inertial forces and contact forces are reflected according to the human-object interaction.

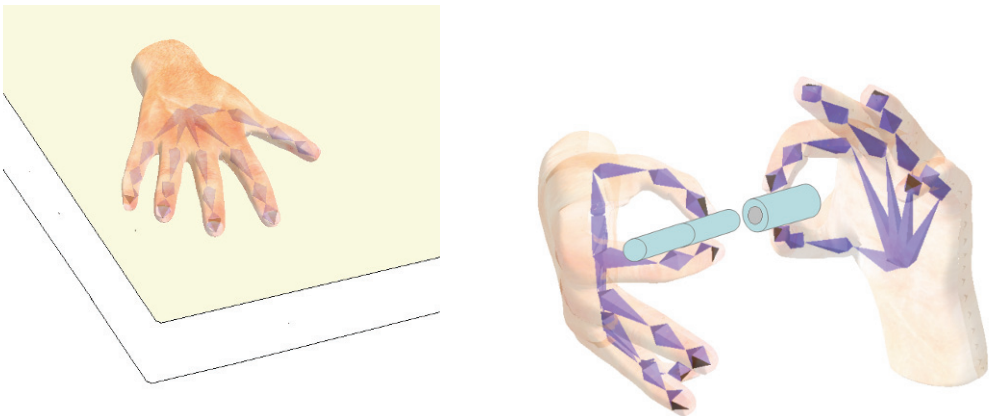


Figure 4.3 Some examples of hand interaction. Fingertips are in contact with a surface (left) and assembling parts (right). Forces are calculated according to task equation description and hand movements.

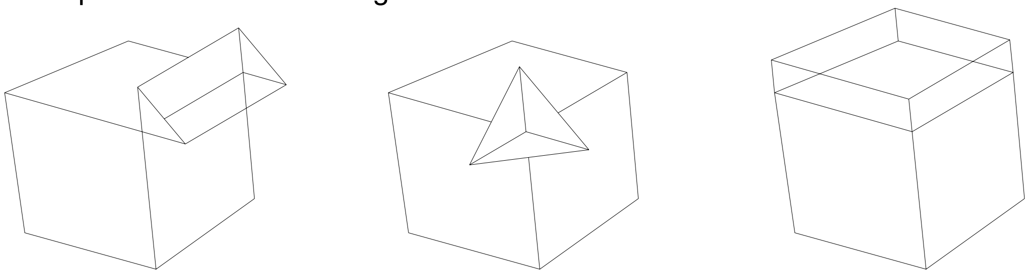
4.3 Elastic forces in contact points

Forces in contact points are considered as elastic. These forces can be determinate as a collision force between two bodies that are in contact implies to resolve a collision detection problem. Collision detection is fundamental to many varied applications including computer games, physically based simulation, robotics, virtual prototyping, engineering simulations, etc. Some applications, such as path planning and animation rendering, do not require real-time performance. Other applications have extraordinary demands on real-time efficiency of collision detection, for instance: haptic (force feedback) systems, particle simulations, surgical simulators, and other virtual reality simulations.

Collision detection concerns the problems of determining if, when, and where two objects come into contact. If involves establishing a Boolean result, answering the question whether or not the object intersect. When must additional determine at what time during a movement collision occurred. Where establish how the objects are coming into contact.

For our purpose we considered only the following geometries: plane, sphere, cylinder, and cube. The geometry of a polyhedron is given in terms of features. The features of a polyhedron are its vertices, edges or faces.

To detect collision between polyhedrons we use the Voronoi theorem, it is based in the concept of Voronoi region [18]. For a feature a Voronoi region is a set of points exterior to the polyhedron which are closer to that feature than any other. The Voronoi regions form a partition of space outside the polyhedron according to the closest feature. The collection of Voronoi regions of each polyhedron is the Voronoi Diagram of the polyhedron. Note that the Voronoi diagram of a convex polyhedron has linear size and consist of polyhedral regions. A cell is the data structure of a Voronoi region. The constraint planes of a Voronoi region are the planes which bound the region. Two adjacent Voronoi regions share a constraint plane. If a point lies on a constraint plane, then it is equidistant from the two features which share this constraint plan in their Voronoi regions.



a) b) c)

Figure 4.4 Three types of Voronoi feature regions of a 3D cube. a) An edge region. b) A vertex region. c) A face region.

Voronio Theorem

> Given non-intersecting polyhedron A and B, let $\mathbf{a}$ and $\mathbf{b}$ be the closest points between feature F_a of A, and feature F_b of B, respectively. If $\mathbf{a}$ and $\mathbf{b}$ are the closest point between A and B, then $\mathbf{a} \in V(F_b)$ and $\mathbf{b} \in V(F_a)$.

For example, consider two polygons *A* and *B* (showed at the left in Figure 4.3) the Voronoi theorem implies $\mathbf{a}$ and $\mathbf{b}$ are the closest points between *A* and *B*. At the right side of $\mathbf{a} \notin V(F_b)$, and so $\mathbf{a}$ and are no longer closest point.

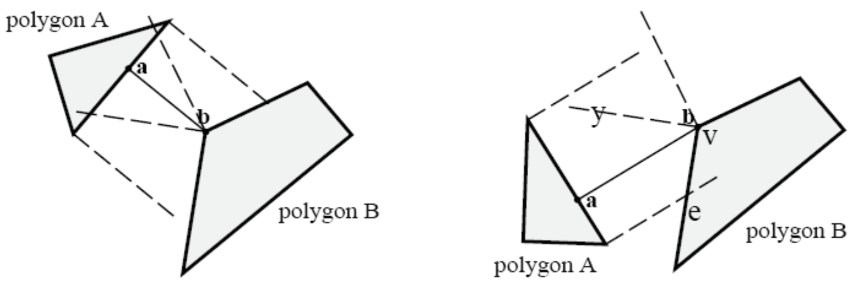


Figure 4.5 Voronoi Regions

The collision force between two objects is modelled in a simple way by means springs at the points of contact. The collision force is computed by means of the hook' law:

$$\mathbf{F} = k d_{\max} \mathbf{x} \tag{4.9}$$

where k is the spring constant, $d_{\max}$ is the maximal penetration distance, and $\mathbf{x}$ is a unit vector which points along the distance that connects the points of maximal penetration.

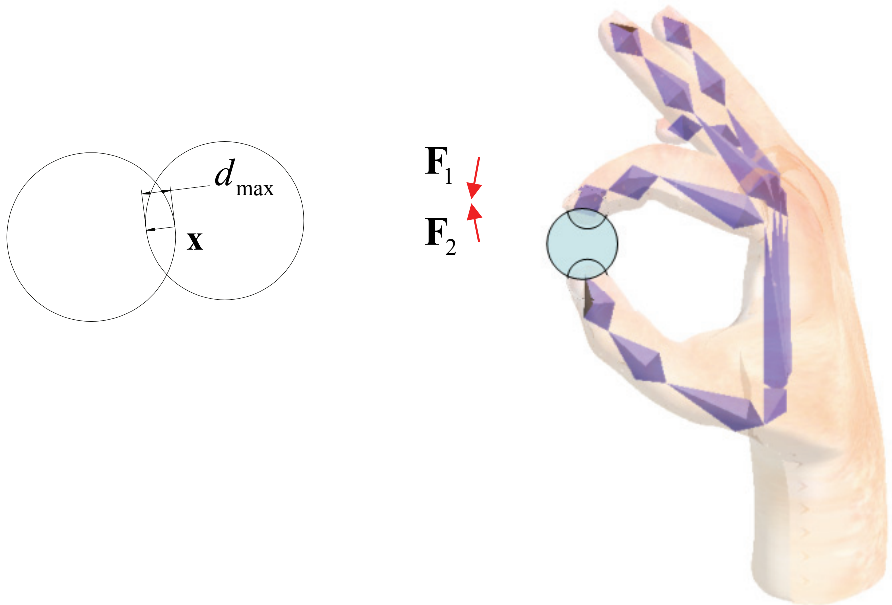


Figure 4.6 Objects in contact and maximal penetration

4.4 Grasping constraints

Forces and torques produced in virtual object require to be modelled so as to create a realistic simulation of grasping in P2O and POP scenarios. The model of multi-finger grasping can be adopted from the analysis of multi-fingered robotic hands [23],[22] to appropriately describe the effects of the whole hand manipulation. [23] To describe the interaction of a multi-fingered hand with an object, a mapping between the fingertip forces and the resulting wrench (forces and torques) on the object with regards to the center of mass are essential.

In order to obtain stable grasp, the center of mass is essential for obtaining equilibrium in torques and forces that take part among fingers and objects. In several studies over quality in prehension have been implemented principally in robotics hands with two or three fingers. In coplanar prehension, it would be best for contact points to be distributed in a uniform manner over object surface so as to improve stability of prehension. An index to quantify this desired uniformity is compared to that of remote internal degrees of polygon prehension whose vertex are contact points of the object.
The effect of inertial and gravitational forces over prehension is minimal when the distance between center of mass CM of the object and the center of contact polyhedron is reduced. Thus, this distance is used as basis of quality measurement for 3D objects.

Power grasp, whose contact points over the object are polygon prehension, is a pentagon of prehension formed by all fingertips. Precision grasp that is generally formed by the thumb, index and middle fingers, the polygon prehension is a triangle prehension. In two-finger precision grasps, the center of mass is midway between the fingers and the plane of two contact surface. Force vectors from fingertips that take part in the contact points of a rigid object when a grasp is produced would be modeled by using contact wrench $F_{c_i} \in \Re^{6\times p}$. The number of fingers in contact can be described by the corresponding wrench basis $B_{c_i} \in \Re^{6\times p}$

where maximum value of p is equal to 5 when all fingertips are in contact. Equation (4.10) shows wrench with points in contact with friction.

$$F_{c_i} = B_{c_i} \cdot f_{c_i} \tag{4.10}$$

The wrench basis for all fingertips in contact is described in equation (4.11), and for soft contact [10]; [31] in equation (4.12). [22]. A soft contact allows beside contact forces, a torque around a normal direction to contact zone. Vector $f_{c_i} \in \Re^p$ describes forces and torques applied by the fingers that correspond to the Coulomb friction model

$$F_{c_i} = B_{c_i} \cdot f_{c_i} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}^T \cdot f_{c_i} \tag{4.11}$$

$$F_{c_i} = B_{c_i} \cdot f_{c_i} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}^T \cdot f_{c_i} \tag{4.12}$$

Friction cone FC models the range of allowable contact forces that can be applied. Coulumb friction model could be considered as defining normal force being positive and lateral forces being proportional to the applied normal force.

Transformation of fingertip forces of multiple (k) contacts into the resulting wrench on the object with respect to the center of mass is described by the contact grasp map $G_i \in \Re^{6\times p}$ as equation (4.13):

$$F_{oc_i} = \begin{bmatrix} R_{oc_i} & 0 \\ \hat{P}_{oc_i} R_{oc_i} & R_{oc_i} \end{bmatrix} \cdot B_{c_i} \cdot f_{c_i} = G_i \cdot f_{c_i} \tag{4.13}$$

The matrix $\hat{P}_{oc_i}$ represents anti-symetrical matrix of the vector p_{oc_i} describing the position of the contact point. Resulting wrench of *k* fingers is described as the sum of contributions from all contact points as described in equation (4.14).

$$f_0 = G_{c_1} \cdot f_{c_1} + G_{c_2} \cdot f_{c_2} + \ldots + G_k \cdot f_{c_k} = G_i \cdot f_{c_i} \tag{4.14}$$

Equation (4.19) defines transformation of matrix of fingertip forces $f_c \in \Re^{kp}$ into the resulting force and torque on the object.

Velocities and forces of fingertips are relationship with torques and velocities of the articulations of the fingers through Jacobean hand $J_h = diag[J_1, \ldots, J_i] \in \Re^{nr\times nm}$. Where $J_i \in \Re^{r\times m}, i = 1, \ldots 5$, is the Jacobean of finger *i* that relates variables in articulation with variables in the fingertip. Equation (4.15) shows the J_h for all fingers. Equations (4.16) and (4.17) show relationship between forces and velocities.

$$J_h(\theta,x_o)=\begin{bmatrix} B_{c_T}^T Ad_{gs_T c_T}^{-1} J_{s_T f_T}^s(\theta_{f_T}) & 0 & 0 & 0 & 0 \\ 0 & B_{c_{IF}}^T Ad_{gs_{IF} c_{IF}}^{-1} J_{s_{IF} f_{IF}}^s(\theta_{f_{IF}}) & 0 & 0 & 0 \\ 0 & 0 & B_{c_{MF}}^T Ad_{gs_{MF} c_{MF}}^{-1} J_{s_{MF} f_{MF}}^s(\theta_{f_{MF}}) & 0 & 0 \\ 0 & 0 & 0 & B_{c_{RF}}^T Ad_{gs_{RF} c_{RF}}^{-1} J_{s_{RF} f_{RF}}^s(\theta_{f_{RF}}) & 0 \\ 0 & 0 & 0 & 0 & B_{c_{LF}}^T Ad_{gs_{LF} c_{LF}}^{-1} J_{s_{LF} f_{LF}}^s(\theta_{f_{LF}}) \end{bmatrix} \tag{4.15}$$

$$T = J_h^T f \tag{4.16}$$

$$v = J_h \dot{\theta} \tag{4.17}$$

A relation between fingertips and net forces generated being applied on the object and a relation between velocity in the points of contact and velocity generated are as follows;

$$J_h(\theta,x)\dot{\theta} = G^T(\theta,x)\dot{x}_o \tag{4.18}$$

If two spheres of unit radio is considered, one for velocity domain and another for force domain, equations (4.18) and (4.19) transform the spheres from ellipsoids to generated velocity and force domain respectively.

$$\dot{x} = H\dot{\theta} \tag{4.19}$$

$$T = H^T\omega \tag{4.20}$$

Ellipsoid velocity is shown in equation (4.21) and ellipsoid force is shown in equation (4.22)

$$\dot{x}^T = (HH^T)^{-1}\dot{x} \le 1 \tag{4.21}$$

$$\omega^T = (HH^T)\omega \le 1 \tag{4.22}$$

Ellipsoids are defined by matrix $(HH^T)^{-1}$ and (HH^T). They have same values and eigenvectors. Principal axis of two ellipsoids coincides but their lengths are inversely proportional. Direction with maximum relation to velocity transmission has the same force transmission relation and vice versa.

Finally, realistic grasping of the hand model should guarantee resistance before external perturbations [7] dexterity, equilibrium and stability [16]; [15]. For realistic interaction with objects in scenarios P2O and POP mathematical model of body dynamics is needed, describe in future work.
The manipulation realistic of the object should keep in current criteria to measure the quality of the prehension under different considerations based on the location of the points of contact on the object or in the configuration of the press element
The localization of the points of contact on the objects is of three methods. The first is based on geometric proprieties of map grasp G [17]; [25]; [20]; [19]. The second is based on geometric relationship as polygon prehension, distance between centoid and center of mass [25]; [20]; [27]; [19]; [24]; [30]; [58]. The third considers limitation in the forces of the fingers [9]; [8]; [26]; [21]; [13]; [33]. The latter is useful in determining force sensors of the thimble.

This measurement of prehension quality is described by several researches as [59]; [60]; [32]; [28].
Criteria of press element use minimum singular value of H, volume of ellipsoid of manipulability, number of condition of H, deviations of the degrees of the articulations of the fingers and index of compatibility of tasks.

4.5 Multi-body Dynamics Software Description (MSIM library)

MSIM is a C++ library for simulation of multi-body dynamics systems. MSIM is capable to compute the forward and inverse kinematic problem, and the forward and inverse dynamic problem of any multi-body system. Currently the MSIM interface is C++ based where the user defines the initial condition of the system, the proprieties and number of bodies, the type of joints, the actuators and their motion commands. There are several subjects, who were involved in the development of MSIM, in this section we will describe the most important. A UML Classes Diagram is presented in Figure 4.7.

The linear algebra methods used in MSIM, are those implemented in BLAS and LAPACK libraries. BLAS, Basic Linear Algebra Subprograms, are standardized application programming interfaces for subroutines to perform basic linear algebra operations such as vector and matrix multiplication. They were first published in 1979, and are used to build larger packages such as LAPACK. Heavily used in high-performance computing, highly optimized implementations of the BLAS interface have been developed by hardware vendors such as by Intel as well as by other authors (e.g. ATLAS is a portable self-optimizing BLAS). The LINPACK benchmark relies heavily on DGEMM, a BLAS subroutine, for its performance. LAPACK the Linear Algebra PACKage, is a software library for numerical computing written in Fortran 77. It provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, Householder transformation to implement QR decomposition on a matrix and singular value problems. Lapack95 uses features of Fortran 95 to simplify the interface of the routines.

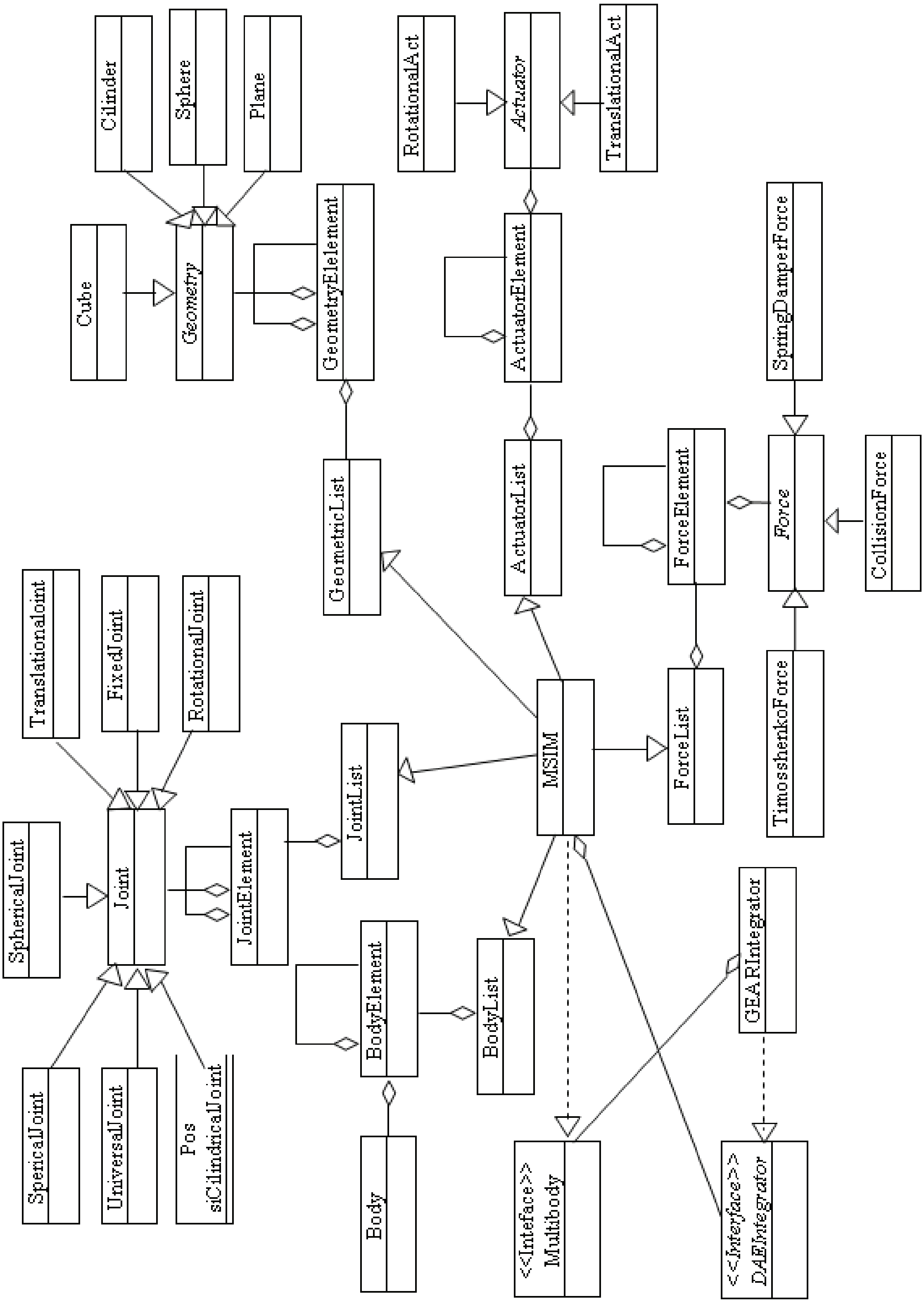


Figure 4.7 UML diagram of MSIM classes

Implementation of MSIM

The matrices used in MSIM are instanced objects of the *double_matrix* class, which encapsulates neccesary data to be used by the developed wrappers of BLAS and LAPACK. The member variables of double_matrix are

- *data.* Which contain the memory address of an array of data corresponding to the vector or matrix.
- *rows.* It refers to the rows of the vector or matrix.
- *columns.* It refers to the columns of the vector or matrix.
- *deletedata.* Specify whether the destructor has to delete data.

Dynamic Kernel

The MSIM dynamic kernel is the component that computes the forward dynamics of a given multi-body system. In order to accomplish it, MSIM groups the multi-body system data in four basic components: *actuators, joints, bodies and forces*. There are four Lists, one for each type of component, that contain all the elements in a sequential form. Components of the Dynamic Kernel are the following:

- Body Class

The basic element of the Dynamic Kernel is *Body* class. This class encapsulates the behaviour of a body, and contains information such as its position and orientation, linear and angular velocities, mass, inertia, and forces that acts on the body (i.e., weight, coriollis force.) Also it contains the angular velocity skew-symetric matrix, the Euler parameter constraint vector, the Euler parameters Jacobian matrix, and the acceleration in terms of Euler parameters.

-Joint Class

The Joint Class is a virtual pure class which defines the behaviour of Joint. Joint Class contains information such as pointers to two bodies that it connects number of restrictions, Jacobian of the joint, acceleration vector and reaction forces produced by the restrictions. All kinds of joints are derived classes of Joint class. Currently there are 6 types of specialization of Joint Class

- FixedJoint Class
- TranslationalJoint Class
- RotationalJoint Class
- CylindricalJoint Class
- UniversalJoint Class
- SphericalJoint Class

-Actuator Class

All the actuators are defined by the virtual pure class Actuator. In MSIM there are to types of actuators: Rotational and Prismatic. A class Joint contains information such as pointers to two bodies that it acts. The Actuator methods permits compute the desired position velocity and acceleration of the actuator, compute the force in the actuator, the vector of driven constraints and its respective Jacobian matrices. Currently there are 2 types of specialization of Actuator Class:

- RotationalActuator Class
- TranslationalActuator Class

-Force Class

The Force class is a virtual pure class, it contains information such as: Number of the force, pointers to the two objects it acts with, and the position where it acts. Its methods compute the vector of external force and torque. There are 3 specialisation classes:

- SpringDamperForce Class
- TimoshenkoForce Class
- CollisionForce Class

List design

There are four lists in MSIM that compose the multi-body system. These lists shares a common feature, they organize the elements in a sequential form. There is a basic structure which consists of a pointer to an element of the list and other pointer which points to a structure with the same number of components. The list has a first element named Root, it is used to begin the iteration over the list. When it is necessary to access to an element of the list, the search begins from the Root element. In Figure 4.6 a UML diagram of the list design is showed.

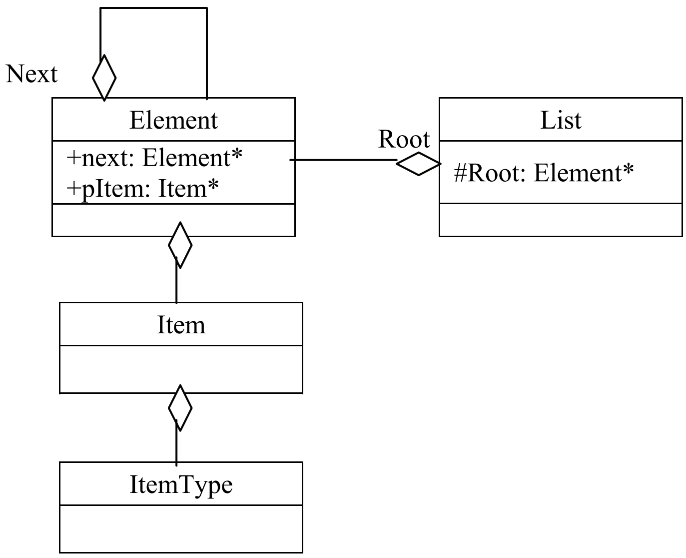


Figure 4.8 UML diagram of List

Integration of the Equation of Motion

The Differential Algebraic Equation (DAE) Integrators used to integrate the equation of motion must be derivate classes from the interface class DAEIntegrator. Currently, the GEAR integrator proposed by Gear et. al. in [12] has been implemented with some modifications. The implemented integrator considers step selection, step size, change the integrator algorithm's order [29].

Modelling a 24 d.o.f. hand in MSIM

A hand of 24 DoF. was modelled in MSIM. The structure of the file where the data was introduced to MISM is showed in Appendices. The model of the hand computes the inverse dynamics problem. For a desired set of joint variables and its respective time derivatives it is possible to compute the required torques at the joints. Since all the rotational joints that compose the hand are active we consider that a rotational actuator was mounted at each one.

The structure of the finger considered here is showed at Figure 4.6. All the fingers consist of 5 d.o.f., except the thumb that consists of 4 DoF. At the last phalange of each finger we consider only a geometric primitive for the collision detection problem

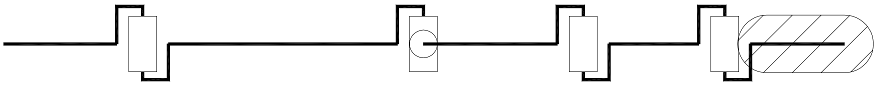


Figure 4.9 Structure of a finger.

This geometry is a cylinder with semi-spheres at the ends. It permits to detect the collisions in a very efficient way. The reason because only the last phalange has attached geometry is that only the last part of the finger is involved in the grasping of the objects.

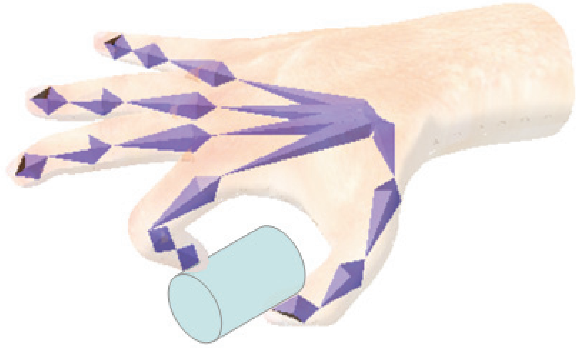


Figure 4.10 Hand grasping a cylinder

In Figure 4.8 a diagram with the flow of data between the project modules shows how the Hand's model (MSIM) interacts with the other components. MSIM model receives the joint variables vector from the Inverse Kinematics module. MSIM computes the inverse dynamics and the collision force between the hand and the objects to be manipulated. Finally the forces data pass to the Master finger to be physically rendered.

Currently, since the Master Finger device only captures the movements of two fingers, the joint variables of the static fingers are set so that the fingers looks like being stretched.

5 Human hand modelling

The kinematics model is done by means of the Denavit – Hartenberg convention used in robotics. Hand model kinematics can be simplified into two finger model kinematics. The first model shows the kinematics model of the index, middle, ring and little fingers with five degrees of freedom (DoF) for each one, and the second represents the kinematics model of the thumb with four degrees of freedom (DoF). These two new models have a different characteristics respect to the firsh DH convention describe in deliverable 4.4.1. This difference consist in moving first the abddution/adduction and then flexion/extension, this difference give a movement more natural to respect to the first configuration described in deliverable 4.4.1.

General finger model

The four fingers have four types of bones: metacarpal, proximal, middle, and distal. The articulations or joints corresponding to these fingers are: Carpometacarpal (CMC) joint, metacarpophalangeal (MCP) joint, proximal Interphalangeal (PIP) joint, and distal interphalangeal (DIP) joint. MCP joints have 2 degrees of freedom of Flexion/Extension and Adduction/Abduction. All of the other joints are flexion/extension. Table 5.1 shows the D-H parameters for the first model.

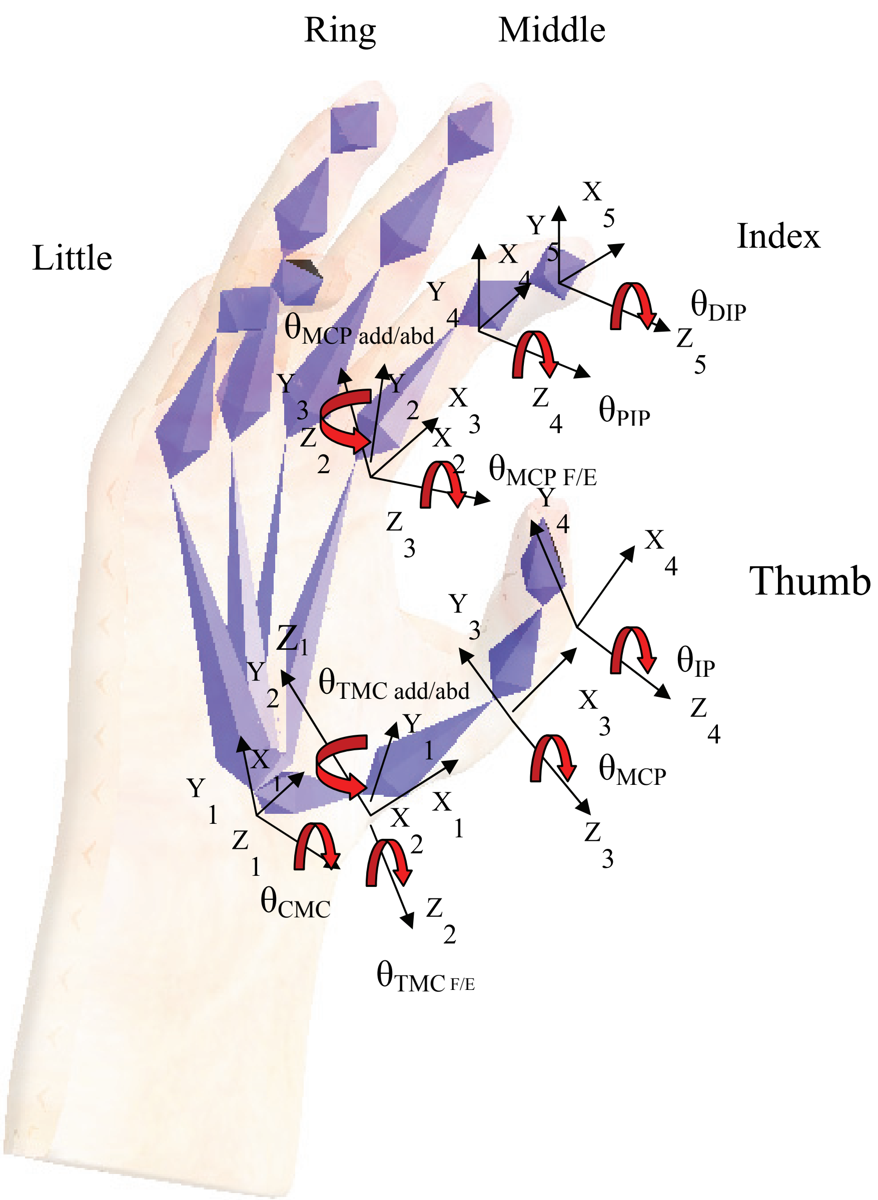


Figure 5.1 Kinematics model

Joint	θ_i	d_i	a_i	α_i
1	θ_{CMC}	0	L1	$-\pi/2$
2	$\theta_{MCP\ Add/Abd}$	0	0	$\pi/2$
3	$\theta_{MCP\ F/E}$	0	L2	0
4	θ_{PIP}	0	L3	0
5	θ_{DIP}	0	L4	0

Table 5.1

Thumb model

The articulations or joints that correspond to this finger are: Trapeziometacarpal (TMC) joint, Metacarpophalangeal (MCP) joint and Interphalangeal (IP) joint. The TMC joint have 2 degrees of freedom. TMC joint have 2 degrees of freedom of Flexion/Extension and Adduction/Abduction Table 5.2 shows the parameters of D-H for the second model.

Joint	θ_i	d_i	a_i	α_i
1	$\theta_{TMC\ Add/Abd}$	*0*	*0*	$\pi/2$
2	$\theta_{TMC\ F/E}$	*0*	*L1*	*0*
3	$\theta_{MCP\ F/E}$	*0*	*L2*	*0*
4	θ_{IP}	*0*	*L3*	*0*

Table 5.2

In Figure 5.1 is shown the kinematic model of the human hand. This figure represents the DH parameter in tables 5.1 and 5.2.

5.1 Constraints

In previous deliverable [4.4.1], several types of constraints were described as rank movement, inter-finger, and intra-finger. In this deliverable, new constraints have been implements in inter-finger and intra-finger so as to acquire realistic movement of the hand model.
Intra-finger constraints have been developed to reproduce movements within of the rank of grip trajectories such as circular and prismatic grasps.
Inter-finger constraints have been verified in experiments carried out with a Cyberglove by Virtual Technologies to obtain the dependency of tendons mainly among the middle, ring and little fingers.

Intra-finger Constraints

In this section, intra-finger constraints are divided into two categories, the first is for index, middle, ring and little fingers, and the second for thumb model.

General finger model

Intra- finger constraint $\theta_{DIP} \approx \frac{2}{3}\theta_{PIP}$ has been accepted by several researches as [38] and efficiently checked in our experiments. However, the constraint for $\theta_{PIP} \approx 2\theta_{MCP(f/e)}$ in experiments carried out had been inappropriate. It is because θ_{PIP} and θ_{DIP} depends on variation of $\theta_{MCP(f/e)}$. The trajectory of the finger with this constraint becomes very close. Consequently, is proposing a new constraint for θ_{PIP} and $\theta_{MCP(f/e)}$ as equation (5.1).

$$\theta_{pip} \approx \frac{3}{4}\theta_{MCP(f/e)} \tag{5.1}$$

Thumb model

The type of constraints being presented has been developed for θ_{ip}, $\theta_{MCP(f/e)}$ and $\theta_{TMC(f7e)}$. The thumb model uses such constraints as described in [4.4.1]. The following equations (5.2) and (5.3) show the intra constraints for thumb model.

$$\theta_{IP} \approx \frac{1}{2}\theta_{MCP(f/e)} \tag{5.2}$$

$$\theta_{MCP(f/e)} \approx \frac{5}{4}\theta_{TMC(f/e)} \qquad (5.3)$$

Inter-finger Constraints

The inter-finger constraints were obtained by using the hand model and Cyberglove. These types of constraints are copious movements among index, middle, ring and little finger. The relationship among angles with middle, ring and little finger has been measured to represent real movements of the hand model.

Cases of coupling movements

There exist coupling copious movement when there is flexion in $\theta_{MCP(f/e)M}$ and the flexions of index and little finger are equal to zero. When this unique flexion is generated, the flexion of middle finger is equal to the flexion of ring finger as described in equation (5.4). Figure 5.2 shows this movement.

$$\theta_{MCP(f/e)M} \approx \theta_{MCP(f/e)R} \qquad (5.4)$$

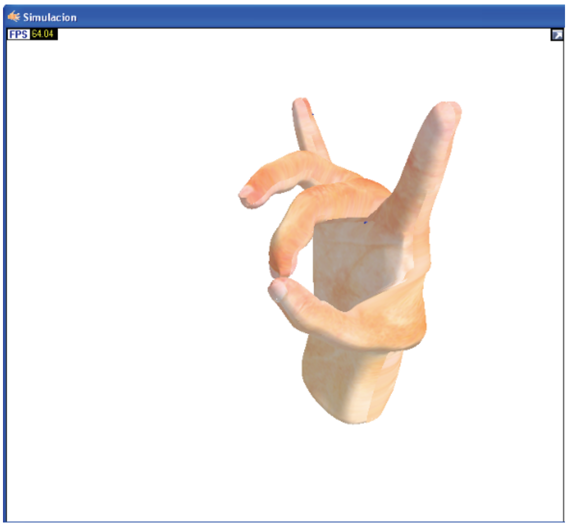


Figure 5.2 Intra finger constraints between middle and ring finger

Another coupling movement is produced when there is flexion solely in the ring finger $\theta_{MCP(f/e)R}$. This causes the flexion of middle finger to be equal to the flexion of little finger as described in equation (5.5) and shown in Figure 5.3.

$$\theta_{MCP(f/e)M} \approx \theta_{MCP(f/e)L} \qquad (5.5)$$

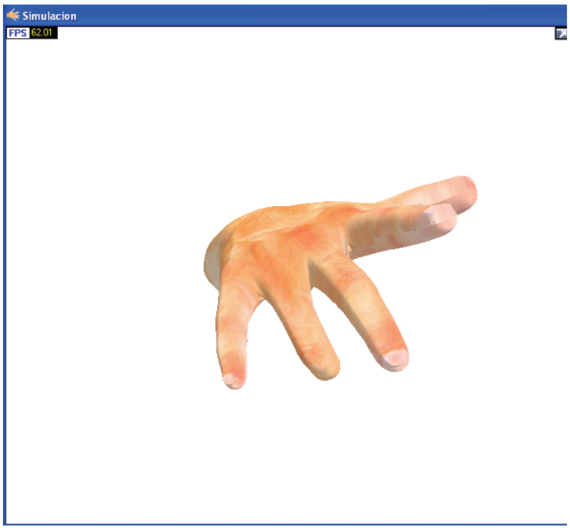


Figure 5.3 Intra-finger constraints among middle, ring and little finger.

Finally, coupling movement in abduction / adduction is genered among ring and little fingers. In most cases, the movement is similar to equation (5.6).and shown in Figure 5.4.

$$\theta_{MCP(abd/add)R} \approx \theta_{MCP(abd/add)L} \tag{5.6}$$

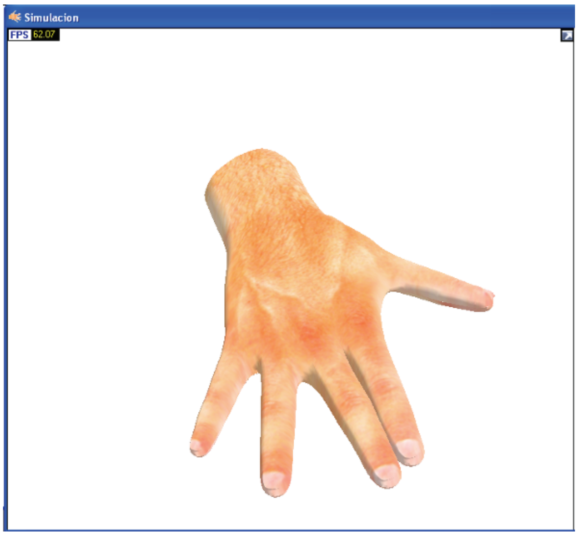


Figure 5.4 Abduction/adduction Intra-finger constraints between the ring and the little finger.

Relations of angles

This kind of relationship is summarized in two types, the most important relationship is when flexion in MCP of little finger exists such as shown in Figure 5.5. Equations (5.7), (5.8) (5.9) and (5.10) represent this type of relationship among these fingers.

$$\theta_{MCP(f/e)R} \approx \frac{7}{12}\theta_{MCP(f/e)L} \tag{5.7}$$

$$\theta_{MCP(f/e)R} \approx \frac{2}{3}\theta_{MCP(f/e)M} \tag{5.8}$$

$$\theta_{MCP(f/e)R} - \theta_{MCP(f/e)M} < 60^{o} \tag{5.9}$$

$$\theta_{MCP(f/e)R} - \theta_{MCP(f/e)L} < 50^{o} \tag{5.10}$$

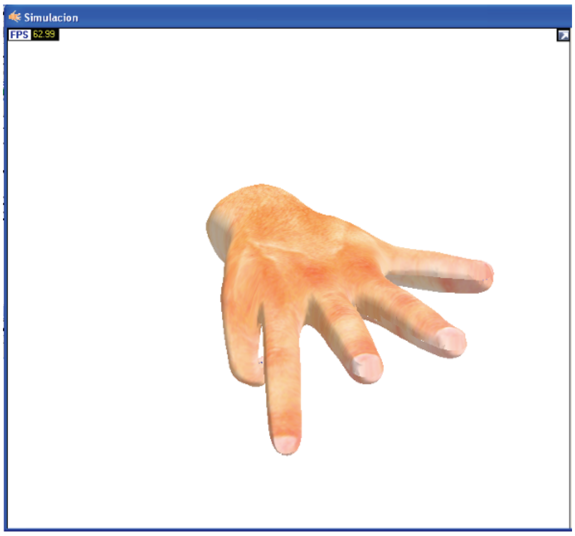


Figure 5.5 Little and ring intra-finger constraints.

The Second relationship is when MCP flexion exists in the index finger such as shown in Figure 5.6. Equation (5.11) represents this type of relationship between the index and the middle fingers.

$$\theta_{MCP(f/e)M} \approx \frac{1}{5}\theta_{MCP(f/e)I} \tag{5.11}$$

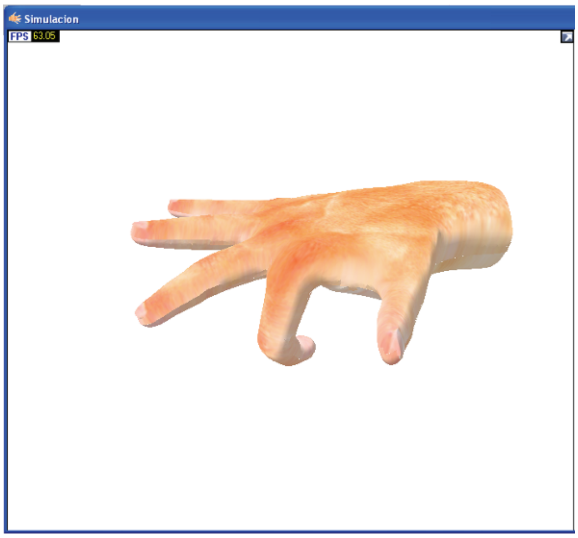


Figure 5.6 Index and middle finger intra-finger constraints

5.2 Inverse kinematics

In serial robot arms, the solution of the inverse kinematics usually is resolved by geometric methods. In this case, the sample relations of triangles are applied in two possible configurations, such as the right arm or the left arm. The human hand can reproduce these types of movements. Movement that is similar to the left or right arm. In such case, the hand becomes flexion/extension & hyperextension. This section describes the possibility of combining this type of movement when inverse kinematics is computed. Direct kinematics n the first general model is described in section 2. Inverse kinematics resolves flexion/extension

& hyperextension of the MCP joint in the first general mode. On the other hand, redundancy that causes abduction/adduction when zero is resolved by a new implementation.
Kinematically, the human hand model is a kinematic redundant. In such case, some model of the robotic hands is solved by means of iterative methods i.e. Newton-Raphson that use Jacobean matrix [42]; [34]. These methods are inadequate in real time applications because they depend on the number of interactions. In experiments using matlab toolbox of robotics when values of all joints vary over 40 degrees, it is diverges because this method uses a mask when Jacobian matrix is not equal to 6 x 6.
In order to reach inverse kinematics model quickly and efficiently, a new method that solves redundant cases for this kinematic model is proposed.

Decision Tree

Redundant situations exist when any part of the human body is modelled. In this case, abduction/adduction plays an important role. If it is equal to zero, the mechanism behaviour becomes a redundant case. It is so, due to flexions/extensions are moved on a plane of two DoFs with four degrees of freedom.
Inverse kinematics is easy to solve when the model does not have any redundant case. The solution becomes complex when hyperextension or negative abduction is reproduced in each finger. A decision tree was done in order to solve such cases.
Decision Tree chooses what equation to solve depending on characteristics such as signs or zero values from [n o a p]. i.e. If abduction/adduction is equal to zero, nz, oz, ax, ay, and pz are all zero. Therefore, it implements a library (dll) that includes this tree for resolving the IK.

The method consists of a correct geometric or mathematic method that is used with equation (5.12) when inverse kinematics diverges in redundant cases or in some other cases:

$$\theta_w = \frac{\theta_c}{\pi e_\varphi} \pm e_\tau \pi \tag{5.12}$$

Where
e_φ. is the mistake in rotation.
e_τ is the mistake in translation
θ_c is the degree calculated from mathematic or geometric method.
θ_w is the degree

If there is only mistake in rotation equation (5.13) is:

$$\theta_w = \frac{\theta_c}{\pi e_\varphi} \tag{5.13}$$

If there is only mistake in translation equation (5.14) is:

$$\theta_w = \theta_c \pm e_\tau \pi \tag{5.14}$$

Description of the method

When computing the value of θ_c , the following procedure calculates values of e_φ and e_τ. The procedure consist of designating minimum and maximum values, depending on rank of desired values, the second passes to obtain e_τ with the minimum value of θ_w .Finally, to obtain e_φ with the maximum value of θ_w

In Figure 5.7, the green curve is from values with mistake in translation. This curve is produced when cmc joint is calculated and produce this translation when abduction/adduction is negative. For cmc joint of the general finger model, solutions until 20 degrees for the little finger can be obtained. The blue curve shows solution until 180º. Blue curve represents convergent solutions until 180º. This curve is obtained using equation (5.14).

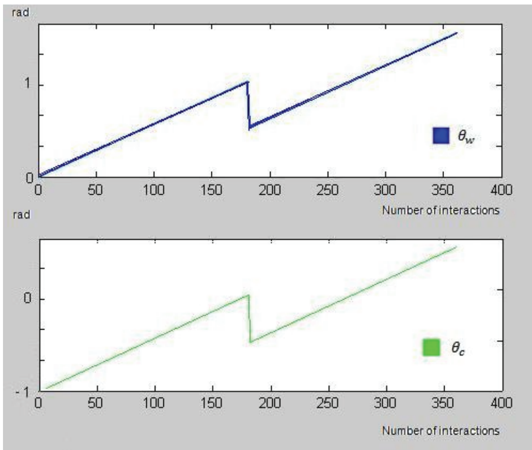
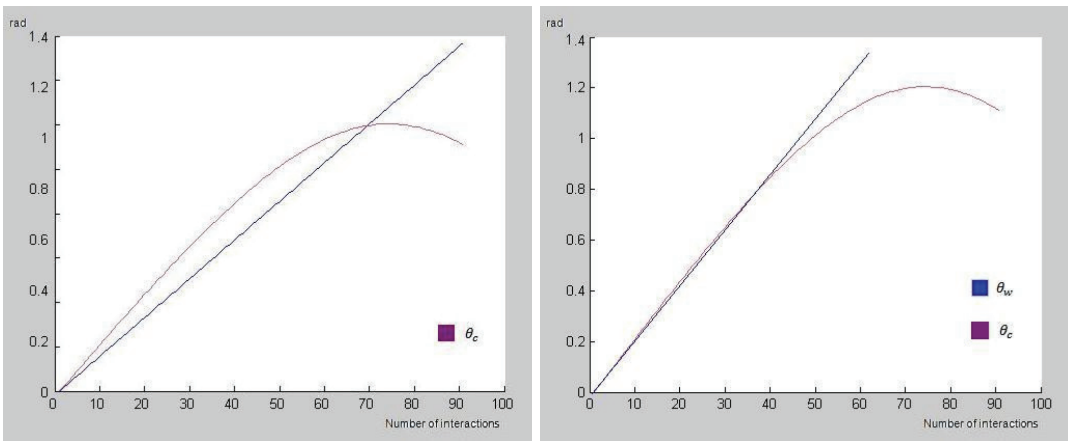


Figure 5.7 Mistake in translation

In Figure 5.8a, the purple curve is the value with rotation mistake. Figure 5.8b is the curve that was corrected with equation (5.13). i.e. for MCP joint, when abduction/adduction is negative as in the curve in Figure 5.8a by geometric methods. Figure 5.8b is the converging solutions obtained at 37º while correcting the mistake in rotation. i.e if there is a maximum adduction in index finger, and then the mcp is flexed, it is impossible to lower it to more than 30º flexion. This relation was obtained by doing a plot with this method.



(a) (b)

Figure 5.8 a) Mistake in rotation. b) Correction in rotation

5.3 Hand Reconstruction

The human hand model has 24 DoFs. This quantity of degrees of freedom requires a longer computational time for several sub-processes, in gesture identification or in computation of the multi-body dynamics in real time .Therefore, it is necessary to reduce the number of elements that make up the original vector from 24 elements to fewer elements.
This section describes a technique for reconstructing the hand model from a descriptor constituted by minimum elements to build the hand model. This descriptor has been obtained from several experiments. This experiment consists of generating grip trajectories that use all constraints as described in section 4. The descriptor is made up of 8 elements that represent minimal information required for generating grip trajectories of the hand model as shown in Table 5.3.

Thumb	Index finger	Middle finger	Ring finger	Little finger
θ_{IP}	θ_{DIP}	θ_{DIP}	θ_{DIP}	θ_{DIP}
$\theta_{TMC(abd\,/\,add)}$	$\theta_{MCP(abd\,/\,add)}$			$\theta_{MCP(abd\,/\,add)}$

Table 5.3 Minimal information in order to represent the hand reconstruction

The descriptor made up of elements from Table 5.3 has been developed in order to obtain grip trajectories for doing circular and prismatic grasps. This descriptor has a characteristic that uses the final joint of the finger (DIP for the finger and IP for the thumb). Thus, the first joint obtained from master-finger would be used in obtaining the following joints applying the intra finger constraints. In this first descriptor, Abduction/Adduction of the middle finger is considered as zero and the Abduction/Adduction of the ring finger is approximately equal to the Abduction/Adduction of little finger. Figure 5.9 represents the reconstruction of the hand using the descriptor 1 with elements above mentioned in Table 5.3.
In gesture recognition, 8 eigenvalues of descriptor 1 are used in building the rest of the 16 eigenvalues to obtain eigenvectors of 24 eigenvalues using the constraints above, and to apply principal component analysis (PCA) to determine the principal components into space (x, y) [35] & [41].

In future collaboration with UNIPI partners for work package 3, collaboration in testing these relations with the development of the glove will be done.
.

Figure 5.9 Reconstruction of the hand model used the descriptor with 8 elements.

Another configuration that consists of using two degrees of MCP joint, descriptor 2 is similar to descriptor 1 in other degrees. Table 5.4 shows part of descriptor 2.

Thumb	Index finger	Middle finger	Ring finger	Little finger
θ_{IP}	$\theta_{MCP(f\,/\,e)}$	$\theta_{MCP(f\,/\,e)}$	$\theta_{MCP(f\,/\,e)}$	$\theta_{MCP(f\,/\,e)}$

$\theta_{TMC(abd/add)}$	$\theta_{MCP(abd/add)}$			$\theta_{MCP(abd/add)}$

Table 5.4 Minimal information for descriptor 2.

Descriptor 2 could be used in experiments with glove developed by UNIPI for optimizing an adequate among dip pip and mcp in situations in which a grip produce grasps with hyperextension in MCP for index, middle, ring and little fingers.

Another future collaboration with UMIPI could consist of performing experiments with respect to the Thumb model in order to obtain correct measurement with respect to degrees of freedom of the thumb model, especially in MCP f/e and TMC abduction/adduction.
Finally, a problem presented in the experiments with Cyberglove by Virtual technologies was to perform measurement of abductions because it has to share a sensor with two sensors. Consequently, it could test using the glove from unipi in order to obtain independent measurement of each finger in abductions/adductions.
The final point of this section is very important because measurement of abduction/adduction in some cases can produce configurations with redundancy in kinematics model when it is approximately zero or when abduction/adduction is negative.

5.4 Implementation of the force from thimble sensors

A real manipulation, proprioceptive and tactile feebback from mechanoreceptor of the fingertips providing information on the grasping force over surface and collision among different objects.
The final objective of hand manipulation consists of carrying out complex manipulations in applications as in scenarios P2O and POP and to provide adequate force feedback.
The thimble of the master finger interface is made up of three sensors from Tekscan Company. These types of sensors are Flexiforce Sensor A201-25.
Normal force in the fingertip and two lateral forces in the fingertip are measured by the integration of three sensors as shown in Figure 5.10.

Figure 5.10 Force sensors of the thimble

In real manipulation, the grip forces are produced when grasping and lifting an object. This manipulation in scenarios can be modeled as a function of the weight of the object. During manipulation, the manipulation dynamics of the object is maintained by the increment or decrement of grip forces and at the same time from tangential forces [39].
When the surface angle on the object increases, normal and tangential forces increase as a function of surface angle. The balance between these forces is however maintained independently from the surface angle, and for which, safety margin remains constant.
Increasing normal and tangential forces are maintained at an approximate constant ratio when an object is being grasped [40].
Forces and tangential torques can be determined from above sensors mentioned into a force domain scenario, master finger interface and a person.

The idea being pursued in force feedback through master finger interface is to provide correct force generation as feedback forces of grasp with objects and obtain precise force measurement at the thimble while performing complex tasks.

6 Haptic modulation: Concepts, haptic enhancing, enhanced haptics, perceptual studies and applications

This task addresses major issues raised in situations where haptic feedback is mixed together from components resulting from different sources. The two reasons why this merging of haptic information becomes necessary are:

- Is it possible and useful to supplement and modulate haptic feedback;
- How to merge direct haptic sensation resulting from touching a real object with the results of simulation, corresponding to the haptic response of virtual objects in the scene.

Prior to answer these questions, we have first to well define the “meaning” of Augmented Haptics or Haptic Modulation and how can it be conceived? Therefore, we introduce a new functional taxonomy dedicated to AH, comparing to what exists for vision based Augmented Reality. Two new concepts are then introduced namely Enhanced Haptics (EH) and Haptic Enhancing (HE). The first category represents the idea that actual virtual or real haptic cues are modified to emphasize, modulate, or to enhance selected haptic information. Haptic enhancing deals with mapping data using haptic cues, providing extra-information with the purpose of guiding the user to achieve a given task. A tele-operated object exploration application and a haptic teaching guidance methodology are presented as instances of respectively the EH and the HE concepts. A possible use to such notions to the P2O and P2P demonstrators is also commented.

6.1 Augmented haptics taxonomy

Classically, AR applications based on vision consist in superimposing graphical information onto a real video. A mixed reality continuum concept has been proposed by Milgram and Kishino [43] where merging virtual and real data is made in a tunable continuum. Because of several well known reasons, AR focused on the visual modality. Hence, existing classifications are mainly inspired from visual requirements. There is no doubt that AR extends to all the other human senses (modalities) [44].

Functional taxonomy

Several taxonomies have been proposed to classify augmented reality. In [45], authors suggested a functional taxonomy. Among the different categories, we retained two of them: *reality with augmented perception or presence*, and *real and virtual combination*.

<u>Visual augmentation</u>

In this first category, virtual or real passive semantic information is added to respectively real or virtual environments. This information consists of keys, symbols, texts... that convey supplementary information to the visual scene of interest. Hence the perception of the reality is enhanced through the extra-information given.

<u>Virtual and real world merging</u>

The second category consists in the merging parts of similar real and virtual worlds in a same display environment. It is the most common situation on virtual reality applications: a virtual model is used to enhance lacking parts of a video real image. Another situation would consist in the reverse path: using real images to enhance realism of a virtual representation. In both cases, a calibration stage is necessary in order that each element superimposes properly. Hence, part of the real/virtual cues can be removed and replaced by its counterparts.

Based on their descriptions, we propose their extension to define haptic augmented reality concepts. There are two reasons why this merging of haptic information becomes necessary:
1. In cases where haptic modality is the main component or when the other modalities, particularly the visual one, are over solicited or not present, it seems logical to use the haptic channel to feed users with extra-information to make them perceive the overall state of the situation.
2. It can be very useful to supplement and modulate haptic feedback (resulting either from simulated or a real source) in order to communicate additional information to the user.

We suggest distinguishing two different concepts: haptic enhancing and enhanced haptic, that we respectively compare to previously addressed vision augmentation and augmented vision, i.e. real/virtual world merging.

Enhanced haptic

We define *enhanced haptic* (or augmented haptics) when haptic modality is amplifying or extrapolating an existing haptic data being fed to the user. The Figure 6.1 illustrates this concept. In scenarios where actual haptic feedback exists, it might be useful to ‘modulate’ the original haptic percept (or data) or to extrapolate what would be missing. This additional transformation of the original haptic data could be seen as what a filter would produce in processing a given picture. It can be convenient to emphasize, erase partially or add new haptic information on top of real haptic components. This is very similar to the vision augmentation category.

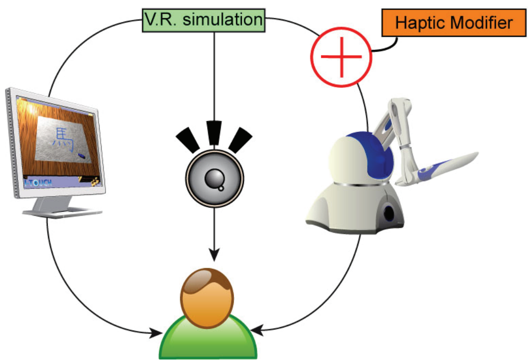


Figure 6.1: Enhanced haptic: the haptic is modified, e.g. amplified/extrapolated/etc.

Haptic enhancing

The haptic enhancing concept (or haptic augmentation) can be considered in scenarios where the haptic modality is solicited to convey extra-information to the user. A way consists in enhancing feedback to the operator by real/actual perception that is not directly available or palpable. It could also be seen as a *corrector metaphor* for demonstration based teaching using an abstraction mapping technique. The source of this extra-information is not necessarily of haptic nature and is not directly related to the current scene or task contexts. Haptic enhancing will blend two different types of haptic information: the first being an actual haptic cue, the second being any information mapped into haptic cues according to a predefined mapping strategy or a mathematic rule. The Figure 6.2 illustrates the idea which is comparable to the virtual and real world merging category of vision-based AR.

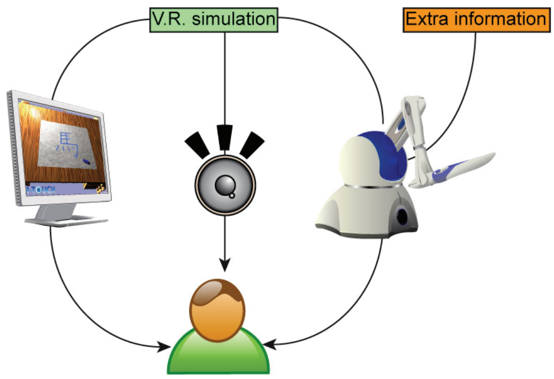


Figure 6.2: Haptic enhancing: a haptic component is added in order to bring extra-information.

6.2 *Enhanced haptic interaction*

Object exploration

As stated in 3.2, it can be convenient to modify the real haptic feedback, resulting from an interaction with an object. We then offer to the operator to explore by touch a given object as fully virtual or fully real or a parameterized blending of both. Hence, from this paradigm, it is made possible to, for instance, either enable a touch diagnosis by comparing the haptic feedback from the interaction with both a real object O_R and its virtual counterpart O_V. To achieve such a goal, we consider two probes, namely P_R the real haptic probe and P_V the virtual one, running in parallel and whose purpose are respectively to sample haptic information h_R from a real object's surface and to compute virtual haptic data h_V from an interaction with a virtual object. In order, to mix both haptic information, returning a unique haptic vector h, we introduce a methodology similar to vision-based AR, namely an blending function B. A linear α-blending instance of B is:

$$h = \alpha \times h_V + (1 - \alpha) \times h_R \quad . \qquad (6.1)$$

where α is a tunable parameter which reflects the blending balance. $\alpha = 1 \rightarrow h = h_V$ and when $\alpha = 0 \rightarrow h = h_R$. α could be a scalar common to each component of $h_{i \in \{V,R\}}$ or a transpose of a vector a in which each component may have a different value of $\alpha \in [0,1]$ interval. Therefore, equation (6.1) is formulated as:

$$a = (\alpha_1, \alpha_2, \ldots \alpha_n)^T \text{ and } h = a^T \times h_V + (1 - a)^T \times h_R \quad . \qquad (6.2)$$

Here, there is no distinction between all the components of $h_{i \in \{V,R\}}$, which can be composed of force, temperature, friction coefficient, etc.

Indirect haptic setup

In order to insure the transparency property of the equation 6.1 and 6.2, an indirect system has to be developed, similarly to vision based AR applications where non-direct optical devices are used. Thus, a tele-operated setup is necessary. The whole teleoperative system developed comprises three major components.

- A standard PHANToM Omni interface, ref Figure 6.3.A, taken as a master haptic device, that measures the user's position in space and that applies feedback forces on the user's hand. Both probes are controlled by the master device.
- A remote *xyz* Cartesian robot taken as a slave robot, equipped with a sensing exploratory probe, ensuring position remote control, as shown in Figures 6.3.B This device has been designed for virtual and augmented reality applications involving multimodal interactions.
- A software implementation designed as a multi-threaded application including:
    - a slave controller implemented to initialize the slave robot configuration and to supervise the position servo-control.
    - a thread dedicated to the VR simulation where a virtual scene of the distant environment is reproduced using OpenGL for the visual display
    - a haptic thread, using the HDAPI library from Sensable Technologies, dealing with the control force feedback mapping.

Working within a local network, the delay is less than 1msec.

Figure 6.3: The whole teleoperated setup comprising a master device (A), a remote slave interacting with real object (B) and a virtual simulation mixing real and virtual haptic and visual data.

Worlds calibration

Since we would like to interact with both real and virtual environments, a calibration procedure is required. Actually, the geometrical transform between the tip of the probe and the explored object is to be known at anytime. Several works are especially dedicated to such an issue as in [46], [47] and [48], and dealing as well with visual occlusion problem.

6.3 Haptic enhancing guidance

Extra-help supplied

Above, we introduced the concept of haptic enhancing which aims at providing extra-information to the user through the haptic modality. We then consider I_U, I_V and I_R to be respectively the information get by the user, the extra-information the used system is made to provide, and the real information acquired in the real world with different sensors:

$$I_U = g(I_R, I_V) \qquad (6.3)$$

g being the function that augment the information. Note that $I = F, \lambda, T, P, \mu, ...,$, which states that the information can either be a force, a luminance, a temperature, etc. Furthermore, I_V can be related to I_R or not. Finally, in some cases where the supplementary information is not directly related to the actual one, *g* can be linear and therefore:

$$I_U = g_1(I_R) + g_2(I_V) \quad . \qquad (6.4)$$

The first part of the equation 6.4 represents the actual haptic cue while the second is the extra-information. When working with the haptic modality, g_1 and g_2 are mathematical functions that map the information $I_{R,V}$ into haptic data and the haptic enhancing concept is written as:

$$H_U = g_1(I_R) + g_2(I_V) = H_R + H_V \quad . \tag{6.5}$$

Training scenario

This help paradigm can be useful in scenarios such as:

- region constraining when using a tele-operated system
- warning the user when moving around in an environment
- path guidance in a training simulation

Similarly to vision-based AR application, we developed a guidance method that is dedicated to help students to learn a path in training simulation such as in the surgical application or handwriting and sport gesture acquisition. Regarding the existing methods for providing help for guidance, such as virtual fixtures, we believe that there are two main drawbacks of such techniques: the passivity of the trainee and a possible dependence arising from the student to the teacher/help. Hence we designed an algorithm in 3 steps that aims at decreasing little by little the haptic help provided, what would result in overcoming those two disadvantages.

This three steps approach is conceived as *a record-and-adaptive replay technique,* which can be used in 2D/3D simulation where haptic is the modularity transferring the extra information. The Figure 6.4 illustrates the different steps. Once a path has been recorded, it is made possible to replay such that little-by-little the intervention of the master (M) is decreasing:

- In the first step, the student (S) is fully driven by the master (full mode).
- Then S is pulled by M on the path, on the second step.
- For the final teaching stage, M help S to come back on the path only when S is lost and is no more on the track.

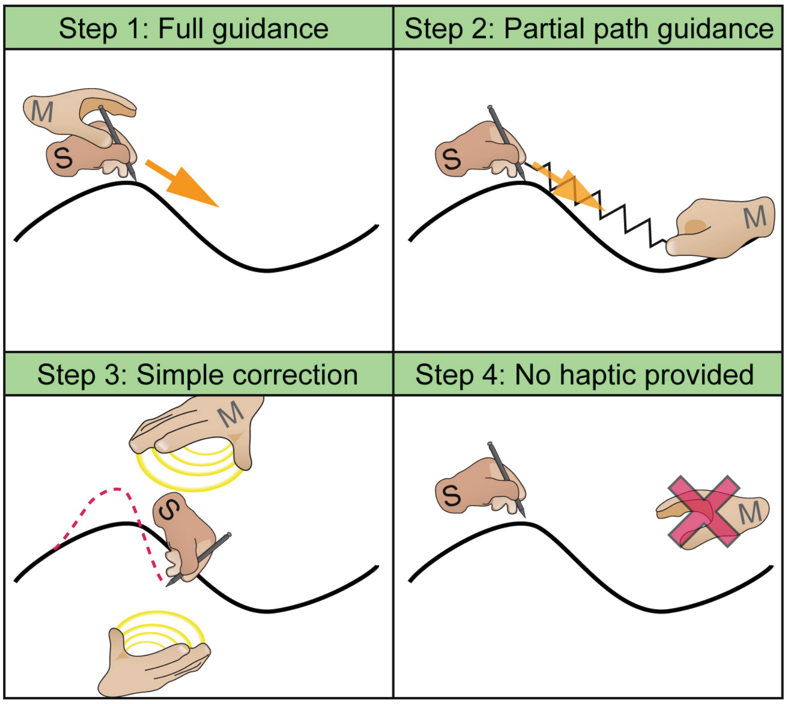


Figure 6.4: The three adaptive steps for teaching, between a master (M) and a student (S).

A first experiment has been done to test the advantages of this adaptative guidance *ag* technique compared to classic haptic guidance, namely fully guided *fg*, using virtual fixtures *vf* or with a simple correction *sc*. The task consisted in learning the path to solve a maze without any visual cues. This last characteristic of the experiment was added so as to avoid any influence of visual learning during the tests. To determine the effectiveness of the methods, we gathered several data including:

- time spent in training
- time taken for going out of the maze in the test session
- covered distance
- total of forces sent back to the user during the training and the test session

Results pointed out advantages of the methodology *ag*:

- Time spent in training is bigger than for the *fg* method but smaller than the two others (*vf* and *sc)*
- Time to resolve the maze is smaller than *fg* and *vf*, and equal to *sc*
- Total of forces sent back during the test session is smaller

The *ag* technique seems to be a compromise that ends in better results.

6.4 Application to the demonstrators

Three threads are used as driving scenarios and testbeds in order to cover the full spectrum of problems arising when implementing interactive multimodal virtual or augmented reality environments; The P2O demonstrator consists in the handling of an object by a human. The P2P one concerns two people engaged in multimodal interaction that involves personal contact (like handshaking) while the POP represents multimodal interaction scenario between two persons mediated by an object.

P2P and POP assistance or skill acquisition

The paradigm for Person-to-Person (P2P) interaction is dance or sports training, where the trainer, for example, teaches a learner through direct limb-to-limb contact. A variant of this scenario is where one of the person is replaced by an entirely programmed virtual character.
For the Person Object Person (POP) demonstrator, two persons are involved in a collaborative task and similarly to the P2P scenario, one of the people could be represented as a virtual avatar. Hence, once the set of data of a given task has been recorded, it would be possible to use it as an input for an avatar behaviour so as to indeed have an interaction, mediated or not, between a person and such a virtual character. As stated previously, the Haptic Enhancing (HE) concept falls into the teaching approach of any kind of application where the haptic modality is prevailing. Thus, the virtual character could be seen as the master teaching the second person how to execute a given task. Looking carefully at the set of data, we select the appropriate ones to modify or overload so that the assignment can be executed faster and more easily. For instance, in the case that two persons try to move an object in one direction, forces sent by the avatar can be emphasized in the direction of movement so that the second person will go along more smoothly.

P2O interaction

The concept of enhanced haptics allows us to modify the features of a real object from the measurement of these values and their modification through the teleoperated setup. The setup does not overcome the hard constraint issue but since it is an indirect interaction, the property can be changed at will. A part of the project tends to record data when people manipulate real objects with their hands. The visual information recording of the object is a first step and is required to build a 3D virtual representation, using the geometrical and texture data gathered. As for the haptic modality, using a sensor, forces, positions, temperatures, and surface roughness at multiple contact points throughout the manipulations can be measured.

Nonetheless the bilateral property of the haptic modality ends in an important set of data which is a limitation to totally recreate a virtual object with similar properties to the real one. Hence, given the exhaustive possibility of interaction with an object, our setup offers the possibility to interact with a virtual object, having the appearance of the real one, using part of the haptic recorded information and compensate missing ones with data coming from a parallel transparent interaction with the real object.
Furthermore, the developed setup and algorithm can become convenient when we want to make the comparison between the people's sensation of interaction with an object; it is indeed possible to make people interact with both virtual and real objects and change from real feedback to virtual, or the way around, in a continuous movement

7 Haptic enhanced avatars

In order to interact with its environment as well as with other avatars/persons, we need an avatar to be given some haptic behaviors: before haptic interaction with objects or people the avatar needs knowledge of this interaction so that it can be haptically driven. We present first a dynamic simulator that will be used with the behavioral knowledge to drive virtual avatar haptic driven interactions. But before a contact occurs it generally has to be decided to interact. Touching the environment to engender motions in an autonomous way is a sub-part of this interaction decision: we present in a second part a contact support planner that addresses this issue.

7.1 Constraint-based interactive dynamic simulation

Introduction

We focused on modeling contact space formulation to be used to drive the dynamic interaction between bodies of different kinematics structure (focusing on virtual avatars) with different materials composing the surrounding environment. Solving contact problems can be made using mainly two methods: the penalty-based method [49] and the constraint-based method. We chose the constraint-based method as non-penetration constraints are explicitly written in the dynamic equations. We succeeded in integrating in contact space formulation altogether the avatar dynamics with other phenomena such as impacts, static and dynamic friction. This work is the extension of the formalism previously introduced by Ruspini and Khatib [50] by including static and dynamic friction without discretizing friction cones, whereas in previous presented approaches, discretization of friction cones is necessary [51], [52], which results in a highly increased computation time. Our main contribution is in efficiently combining the operational space formulation of the multi-body dynamics in the contact space and solving contact forces with friction using an iterative Gauss-Seidel approach that has been applied recently to robotics and to deformable objects [53], [54].

Main algorithm

The problem we have to solve for contacts is described as a complementarity problem:

$$0 \leq f \perp \Lambda^{-1} f + B \geq 0 \tag{7.1}$$

where f is the contact forces, $\Lambda^{-1}=JM^{-1}J^T$ is the projection of the inertia matrix in the contact space and $B = JM^{-1}[\Gamma - b - g] + \dot{J}\dot{q}$ is the free relative acceleration of the contact points that is computed by the forward dynamic model with Featherstone's algorithm [55]. This equation is written without taking into account friction. Adding Coulomb's friction implies to write

equation (7.1) in terms of velocities (otherwise there may be no solution for friction forces). Therefore we integrate equation (7.1) with a simple Euler integration scheme.
To obtain Λ^{-1} we start from the forward dynamic equation of a multi-body system that gives joint accelerations:

$$\ddot{q} = M^{-1}[\Gamma - b - g] + M^{-1}J^T f \qquad (7.2)$$

where *M* is the inertia matrix of the whole system, Γ is the applied torques, *b* is the centrifugal and Coriolis effects and *g* the gravity. Then cartesian accelerations are:

$$a = JM^{-1}[\Gamma - b - g] + \dot{J}\dot{q} + JM^{-1}J^T f = B + \Lambda^{-1} f \qquad (7.3)$$

As said before, *B* is known. We compute again Featherstone's algorithm considering there is no torques (Γ=0), no gravity (*g*=0) and no joint velocities ($\dot{q} = 0$) and setting *f* to unit forces with respect to the normal and tangential directions in the contact space as external forces. Then the free acceleration *B* becomes zero and we get

$$a = JM^{-1}J^T = \Lambda^{-1}$$

Knowing *B* and Λ^{-1} allows solving contact forces using an iterative Gauss-Seidel approach. Note that Λ^{-1} is a square matrix of size 3*m*, with m the number of contact points.
For solving impacts, the method is similar. Technical development details that allow an efficient implementation and problems have been published in IEEE ROBIO 2006 [56].

Results

Comparing to previously presented approaches in this domain, our work shows complex scenarios involving humanoid avatars in manipulation tasks while contacting with the environment (Figure 7.1). We also made a first comparison with the reality by performing experiments (Figure 7.2). We managed to get a computation time of $O(nm+m^2)$ where *n* is the number of bodies and *m* the number of contact points. For a 67-contact point case, we compute a square matrix Λ^{-1} of size 201, whereas using previously presented approaches, this matrix is of size 1734.

Figure 7.1 Multibody dynamic simulator for virtual avatars (humanoid avatar performing complex tasks with multi-contact support)

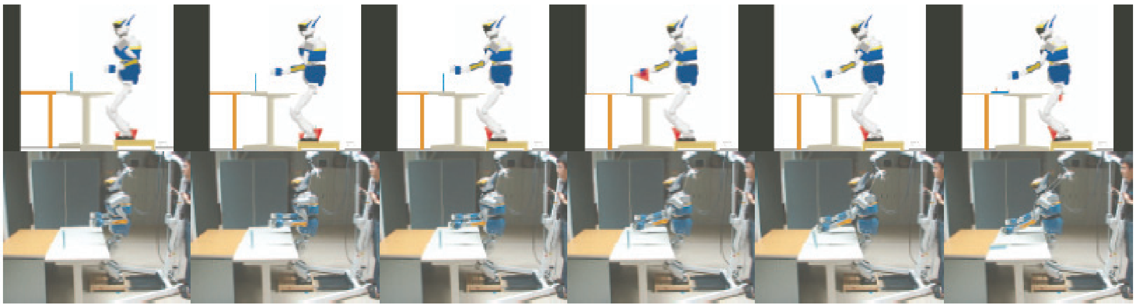


Figure 7.2 Comparison between simulation and reality

Extension of the contact model

In these simulations, we considered only sliding friction. In the reality, pivoting and rolling behaviors can also occur. Thus, we should consider also these friction types in our simulator. We made preliminary studies first on rolling friction.
In simulation, objects are discretized. Depending on the discretization (mesh) of these objects, their dynamical behavior can change. For example, considering two cylinders of the same size but differently discretized and keeping the same contact model, we give them the same initial angular velocity and see how they move on a flat surface. The differences are reported in Figure 7.3.

We clearly see that the more discretized the cylinder is, that means the smaller the contact surfaces are, the less energy it will loose and so the more it will roll. This can also be seen in the reality. We also see that when nearly stopping rolling, the cylinders start bouncing, but this bouncing depends on the discretization and is reduced when highly discretized. However this is not a realistic behavior, since in the reality round-shaped objects do not bounce when stopping rolling. Moreover, in simulation the precision of discretization is limited, meaning that with the present model, we will always have bouncing. Therefore, we should treat rolling behavior in a different manner. We will define an appropriate model of rolling friction. We will also integrate an appropriate model of torsional friction.

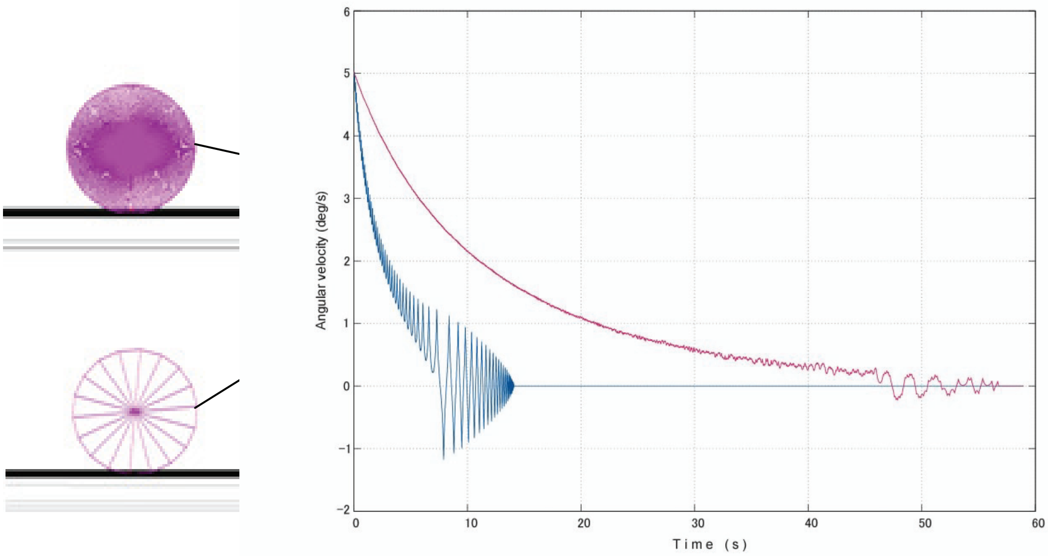


Figure 7.3 Influence of discretization of the cylinder on the dynamical behavior (in blue: with 18 edges, in pink: with 180 edges)

7.2 Contact support planer

Introduction

Nowadays, most of the planning algorithms relative to virtual humanoid avatars suffer from two major drawbacks: they usually focus only on walking motions or/and are designed to avoid obstacles. Contacts are taken by the feet only, while the rest of the avatar is asked not to touch the environment. This implies the avatar performs only a small subset of the movements it is theoretically able to achieve.
In order to improve this range of movements, we aim at making an avatar use every parts of its body to generate a movement while maintaining its stability. A good example, where such a planning is needed, is shown in the Figure 7.4: the operator places an avatar in front of a table and asks it to grab a can that was put on this table, but out of reach. The avatar needs to go toward the table and take support contact points on it, with a hand and a leg, to reach the can while remaining stable.
To achieve such, a first implementation of a contact support planner (CSP) has been done: given an initial posture and a final goal, the planner will find, if it is possible, a trajectory where the avatar interacts with his environment to move toward the goal, by choosing support contact points on it. In particular, this algorithm considers walking motions as a particular type of motion and therefore is a wide generalization of it.

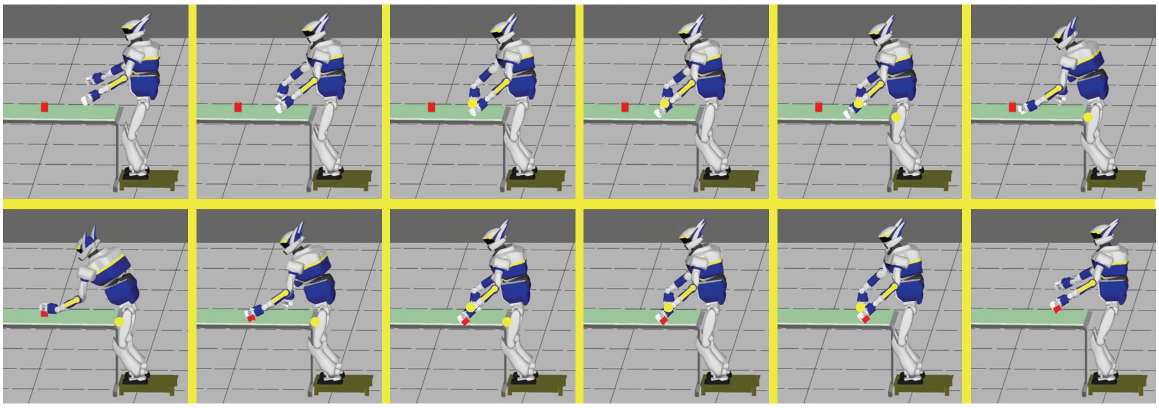


Figure 7.4 Planning support contact point to induce motion of a virtual avatar

CSP Main algorithm

Our algorithm relies on two main parts (Figure 7.5):

- a tree builder/explorer, that handles the planning part and acts in the set-of-contacts space,
- a posture generator, that try and find of collision-free stable posture of the avatar for a given set of contacts.

Tree builder/explorer

The tree builder/explorer (TBE) is the upper part of the planner. It incrementally builds a tree, each node of it being a set of contacts that has been validated by the posture generator (i.e. it is possible to find a posture where the avatar is interacting with the environment as specified by the set of contacts).
The input of the TBE is a set of contacts along with its posture, as well as a set of end conditions; the output is a sequence of postures and associated contacts. At each step the TBE starts from the current node. First it asks the posture generator to find a posture with the current set of contacts but that also comply with the end conditions. On a successful return of the posture generator, the algorithm terminates by returning the path from the initial node to the current node. Otherwise, sons of the current node (i.e. new sets of contacts) have to be generated. It is done in two ways:

- deleting a contact from the current set,
- creating a new between the environment and a part of the avatar that is not already involved in an interaction.

Each new set of contacts is given to the posture generator and kept as a new node whenever a posture is found. New nodes are evaluated according to a planning objective function that is designed to drive the avatar toward the objective. The new current node is the best node generated so far, according to this function (this best node may not be one of the newly generated ones).

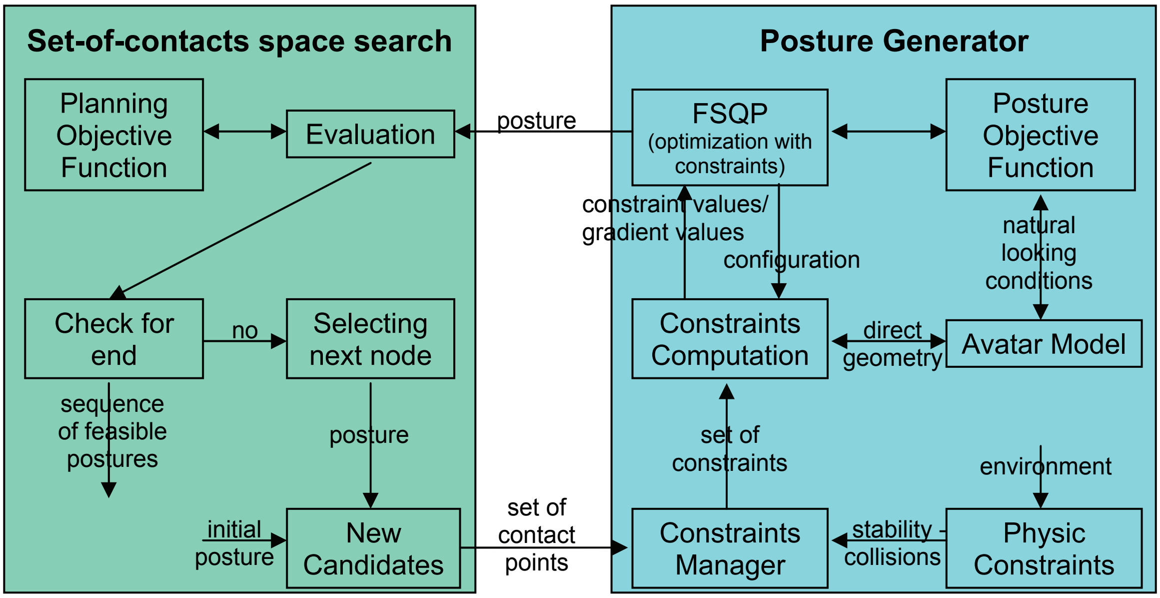


Figure 7.5 Overall architecture of the contact support planner

Posture generator

The Posture Generator can be seen as a kind of generalized inverse kinematics module. Its task is to find a posture of the avatar that respects geometrical and physical constraints:

- contact constraints,
- self-collision and collision constraints
- stability constraints
- …

So far collision constraints are not taken into account: constraints need to be written in a two times continuous way, while distance between objects presents discontinuities of its derivatives. Work on this subject is actually underway and will be presented at M30.

The posture generation is based on FSQP which is a optimization algorithm handling generic smooth objective functions under generic smooth constraints. The main work of our Posture Generator is to generate and translate all constraints for FSQP. The posture objective function is defined by the user and drives the look of the posture. It can for example define a criterion to generate a human-like posture.

Results

So far, some nice planning results have been obtained including the example of Figure 7.4 or pure walking motion extended to the climbing of stairs. Some of these results have been published in IEEE IROS 2006 [57].

8 Conclusions

A basic framework has been developed that generates haptic feedback based on force signals that have been recorded along arbitrary trajectories in a real environment. The conducted research focused on designing methods to generate haptic feedback which is purely based on previously recorded quantities. No models are employed to render the haptic properties of the scene content. This concept provides a framework which is not dependent on any assumptions about the objects that the user interacts with. Besides an algorithm to display quasi static feedback to the user, an extension has been proposed which allows incorporating slip during the haptic replay. The current results are promising, however, several aspects are mentioned how to enhance the capabilities of the framework yet further to improve the quality of the feedback.

The advances in the development of a multi-body dynamic hand model have been presented in sections 4 and 5. A demonstrator of object manipulation has been developed and a realistic human hand model has been defined. Object manipulation demonstrator is based on a multi-body dynamics software library which describes the interaction between a haptic device and a virtual manipulated part. The human hand model has been developed in order to realistically characterize and model hand postures during object grasping.

Concepts for haptic modulation have been presented in section 6. A terminology has been established which distinguish between the pure manipulation of haptic signals by data processing techniques and the augmentation of the raw haptic feedback by signals based on additional information. Furthermore, formulas have been proposed how to merge two haptic signals from two different sources into one signal which is provided to the user. Examples for both strategies have been presented and applications for an implemented demonstrator covering all three scenario classes P2O, P2P and POP have been proposed.

In section 7 a technique has been described that integrates haptic cues like friction into contact space formulations to drive the dynamic interaction of avatars within an environment with different material properties. A constraint-based method has been chosen to solve contact problems. All the avatar dynamics and the haptic cues have been integrated into one contact space formulation while a high performance of the underlying algorithms is maintained. Experiments have been conducted that provide a first comparison with the reality. Furthermore, some planning results have been obtained including pure walking motion extended to the climbing of stairs.

9 References

[1] K. Salisbury, F. Conti, F. Barbagli, “Haptic Rendering: Introductory Concepts”, IEEE Computer Graphics and Applications, 2004

[2] V. Hayward, B. Armstrong, “A New Computational Model of Friction Applied to Haptic Rendering”, Experimental Robotics VI, P. Corke and J. Trevelyan (Eds.), Springer: New York, LNCS 250, 2000, pp. 404-412

[3] V. Hayward, “Haptic Synthesis”, 8th International IFAC Symposium on Robot Control, SYROCO 2006

[4] M.C. Cavusoglu, D. Feygin, F. Tendick, “A Critical Study of the Mechanical and Electrical Properties of the PHANTOM Haptic Interface and Improvements for High Perfromance Control”, Presence, Vol. 11, No. 6, 2002

[5] J.E. Colgate, J.M. Brown, “Factors Affecting the Z-Width of a Haptic Display”, Proceedings of the IEEE International Conference on Robotics & Automation, 1994, pp. 3205-10

[6] A. Bicchi. On the closure proprieties of robotic grasping. Int. J. Robotics Research. 319-344. 2000.

[7] A. Bicchi and V. Kumar. Robotic grasping and contact: A review. In: Proc. IEEE ICRA 2000. 348-352. 2000.

[8] R.C. Brost. And K. Y. Goldberg. A complete algorithm for designing planar fixtures using modular components. IEEE Trans. Rpbotics and Automation. 31-46.

[9] M. Buss, H. Hashimoto and J.B. Moore. Grasping force optimization for multifingered robot hands. In: Proc. IEEE ICRA. 1034-1.039. 1995.

[10] M. Buss, H. Hashimoto and J.B. Moore. Dextrous hand grasping force optimization. IEEE Trans. Robotics and Automation. 406-418. 1996.

[11] Salvador Cobos, Manuel Ferre, Rafael Aracil, M. A. Sanchéz-Urán, and Javier Ortego, “Hand Gesture Recognition for Haptic Interaction”. HCII’ International 2007, 12th International Conference on Human-Computer Interaction (2007).

[12] Gear, Leimkuhler and Gupta. “Automatic Integration of Euler-Lagrange Equation with Constraints”. Journal of comp. and applied math. Vol 12 and 13, pag 77-90. 1985.

[13] L. Han, J.C. Trinkle and Z.X Li. Grasp analysis as linear matrix inequality problems. IEEE Trans. Robotics and Automation. 663-674. 2000.

[14] Haug, E.J., “Computer-Aided Kinematics and Dynamics of Mechanical Systems, Volume I: Basic Methods”, Allyn and Bacon, (1989).

[15] W.S. Howard and V. Kumar. On the stability of graped objects. IEEE Trans. Robotics and Automation. 904-917.1996.

[16] J. Kerr and B. Roth. Analysis of multifingered robot hands. Int. J. Robotics Research 3-17.1986.

[17] Z. Li and S. Sastry. Task-oriented optimal grasping by multifingered robotic hands. In. Prec. IEEE ICRA. 389-394. 1987.

[18] Lin, Ming. John Canny. “A Fast Algorithm for Incremental Distance Calculation,” Proccedings of the 1991 IEEE International Conference on Robotics and Automation, pp. 1008-1014, April 1991.

[19] Y.H. Liu. Computing n-finger form-closure grasps on polygonal objects, Int. J. Robotics Research. 19 (2). 149-158. 2000.

[20] B. Mirtich and J.Canny. Easily computable optimum grasps in 2D and 3D, proc. IEEE ICRA. 739-747. 1994.

[21] B. Mishra. Grasp metrics: Optimality and complexity, Algorithmic Foundations of Robotics (K. Goldberg, d. Halperin, J.C. Latombe and R. Wilson, Eds.). 137-166. 1995.

[22] D.J. Montana. The kinematics of multinfingered manipulation, IEEE Transactions on Robotics and Automation. 491-503. 1995

[23] R. M. Murray, Z. Li and S. Sastry. A Mathematical Introduction of Robotic Manipulation. CRC Press. Boca Ratón, Florida. 1994.
[24] V.D. Nguyen. Constructing force-closure grasps, Int. J. Robotics Research 7 (3). 3-16. 1988.
[25] Y.C. Park and G.P. Starr. Grasp synthesis of polygonal objects using a three-fingered robotic hand, Int. J. Robotics Research 11 (3). 163-184. 1992.
[26] N. S. Pollard. Synthesizing grasps from generalized prototypes, proc. IEEE ICRA. 2124-2130. 1996.
[27] J. Ponce, S. Sullivan, A. Sudsan, J.D. Boissonat and J.P. Merlet. On computing four-finger equilibrium and force-closure grasps of polyhedral objects, Int. J. Robotics Research 16(1). 11-35. 1997.
[28] J.K. Salisbury and J.J. Craig. Articulated hands: Force control and kinematic issues, Int. J. Robotics Research 1(1). 4-17. 1982.
[29] Shampine and Reichelt. “The Matlab Ode Suite”. SIAM J. sci. Computer. Vol 18, No 1. pag 1-22. 1997.
[30] D. Stam, J. Ponce and B. Faverjon. A system for planning and executing to-finger force-closure grasps on curved 2-D objects, proc. IEEE/RSJ IROS. 210-217. 1992.
[31] Xydas, N. And I. Kao. Modeling of contact mechanics and friction limit surface for soft fingers in robotics with experimental results. Int. J. Robotics Research. 941-950.
[32] T. Yoshikawa. Analysis and control of robot manipulators with redundancy, proc. 1st Int. Symposium of Robotic Research. 735-747. 1984.
[33] X. Zhu, H. Ding and H. Li. A quantitative measure for multifingered grasps, proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics. 213-219. 2001.
[34] A. Balestrino, G. De Maria, and L. Sciavicco. Robust Control of robotic manipulators, in Proceedings of the 9th IFAC World Congress. 2435-2440.1984.
[35] S. Cobos, M. Ferre M.A. Sanchéz-Urán and J. Ortego. Hand Gesture Recognition for Haptic Interaction. In Proc. Human Computer Interaction HCII 2007.
[36] M.R. Cutkosky and P.K. Wright. Modeling Manufacturing grips and correlations with design of robotic hands. Procedings of the IEEE International Conference on Robotics and Automation. 1533-1539, 1986.
[37] M.R. Cutkosky. On grasp choice, grasp models, and the design of hands for manufacturing tasks. IEEE trans. Robotics and automation. 269-279.1989
[38] C.S Fahn and H. Sun. Development of a Data Glove with Reducing Sensors Based on Magnetic Induction.IEEE Transactions on Industrial Electronics, vol. 52, No.2. 2005.
[39] P. Jenmalm and R.S Johansson. Visual and somatosensory information about object shape control manipulative fingertip forces. Journal of Neuroscience. 4486-4499. 1997
[40] P. Jenmalm, A. W. Googwing and R.S Johansson. Control of grasp stability when humans lift objects with different surface curvatures. Journal of Neurophysiology. 1643-1652. 1998.
[41] M. Oyarzabal, M. Ferre, S. Cobos. M. Monrroy, J. Barrio and J. Ortego.Multifinger Haptic Interface for Collaborative Tasks in Virtual Environments.In: Proc. LNCS. 673-680. 2007
[42] W. A. Wolovich and H. Elliot, A computational technique for inverse kinematics, in Proc. 23rd IEEE Conference on Decision and Control. 1359-1363.1984.
[43] P. Milgram and F. Kishino. A taxonomy of mixed reality visual displays. IEICE Trans. Information Systems, E77-D:1321 – 1329, 1994.
[44] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. Mac-Intyre. Recent advances in augmented reality. IEEE Comput. Graph.Appl., 21(6):34 – 47, 2001.
[45] P. Fuchs, G. Moreau, and J.P. Papin. Le traité de la réalité virtuelle. In les Presses de l’Ecole des Mines de Paris editor,,Paris, November 2001.

[46] J. Vallino and C.M. Brown. Haptics in augmented reality. In Proceedings of the IEEE International Conference on Multimedia Computing and Systems, pp. 91 – 95, June 1999.
[47] G. Bianchi, C. Jung, B. Knoerlein, G. Szekely, M. Harders, "High-fidelity visuo-haptic interaction with virtual objects in multi-modal AR systems", ISMAR'06, pp. 187-196, 2006.
[48] G. Bianchi, B. Knoerlein, G. Szekely, M. Harders, "High Precision Augmented Reality Haptics", Proc. of EuroHaptics'06, pp. 169-178, 2006.
[49] K. Yamane and Y. Nakamura, “Stable penalty-based model of frictional contacts,” in IEEE International Conference on Robotics and Automation, 2006.
[50] D. C. Ruspini and O. Khatib, “Collision/contact models for dynamics simulation and haptic interaction,” in International Symposium of Robotics Research, 1999.
[51] M. Anitescu and F. A. Potra, “A time-stepping method for stiff multibody dynamics with contact and friction,” International Journal for Numerical Methods in Engineering, 2002.
[52] D. Baraff, “Fast contact force computation for nonpenetrating rigid bodies,” in SIGGRAPH, 1994.
[53] T. Liu and M. Y. Wang, “Computation of three-dimensional rigidbody dynamics with multiple unilateral contacts using time-stepping and gauss-seidel methods,” IEEE Transactions on Automation Science and Engineering, 2005.
[54] C. Duriez, F. Dubois, A. Kheddar, and C. Andriot, “Realistic haptic rendering of interacting deformable objects in virtual environments,” IEEE Transactions on Vizualization and Computer Graphics, 2006.
[55] R. Featherstone, Robot dynamics algorithms. Kluwer Academic Publishers, 1987.
[56] J.-R. Chardonnet, S. Miossec, A. Kheddar, H. Arisumi, H. Hirukawa, F. Pierrot, and K. Yokoi, “Dynamic simulator for humanoids using constraint-based method with static friction,” in IEEE International Conference on Robotics and Biomimetics, 2006
[57] A. Escande, A. Kheddar, S. Miossec, Planning support contact-points for humanoid robots and experiments on HRP–2, IEEE/RSJ International Conference on Intelligent Robotics and Systems (IROS), October 9-15, Beijing, China.
[58] B. Kim, S. Oh, B. Yi and I.H. Suh. Optimal grasping based on non- dimensionalized performance indices. In: Proc. IEEE/RSJ IROS 2001. 949-956. 2001.
[59] K. B. Shimoga. Robot grasp synthesis algorithms: A survey. Int. J. Robotics Research. 15(3) 230-266. 1996.
[60] C. A. Klein and B.E. Blaho. Dexterity measures for the design and control of kinematically redundant manipulator. Int. J. Robotics Research 6(2) 72-83. 1987

10 Appendices