Multibody Dynamics Model of a Human Hand for Haptics Interaction

Hector Moreno, Roque Saltaren, Manuel Ferre, Eugenio Yime, Rafael Aracil, and Isela Carrera

Universidad Politecnica de Madrid
Escuela Tecnica Superior de Ingenieros Industriales, DISAM C/ Jose Gutirrez Abascal, 2 28006 Madrid Spain
{hmoreno,rsaltaren,mferre,eyime,aracil,icarrera}@etsii.upm.es
http://www.disam.upm.es

Abstract. In this paper we propose a strategy for modelling a human hand for Haptics interaction.The strategy consists in a parallel computing architecture that calculates the dynamics of a hand, this is accomplished by computing the dynamics of each finger in a parallel manner. In this approach multiple threads (e.g. haptics thread, graphics thread, collision detection thread, etc.) run concurrently and therefore we developed a synchronization mechanism for data exchange. We describe in detail the elements of the developed software.

Keywords: Multibody dynamics, haptics and parallel computing.

1 Introduction

Haptic interaction can be combined with the dynamic simulation to allow rich, intuitive interactions with virtual environments. This has many applications such as teleoperation of robots for remote inspection, virtual training, ergonomics-based design, etc. Because of these reasons Haptics combined with dynamic simulation is being studied by several researches, [1], [2], and [3]. In [1], Diego Ruspini and Oussama Khatib presented a framework for haptic simulation of multi-body contact dynamics. In the framework, a virtual proxy is attached to a rigid body, so the Phantom can be used to control the motion of a body in a contacted environment.

We modeled the dynamics of a 24 d.o.f. hand previously studied in [4]. 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. All the fingers consist of 5 d.o.f., except the thumb that consists of 4 d.o.f. At the last phalange of each finger we consider only a geometric primitive for the collision detection problem.

M. Ferre (Ed.): EuroHaptics 2008, LNCS 5024, pp. 700–705, 2008.

2 Multibody Constrained Equations in Terms of Euler Parameters

The Newton-Euler equation of motion permits to model the dynamic behavior a multi-body system.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.

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: $\mathbf{p} = [e_0 \quad e^T]^T$ where $e_0 = \cos(\frac{\chi}{2})$ and $\mathbf{e} = [e_1 \quad e_2 \quad e_3] = \mathbf{u}\sin(\frac{\chi}{2})$.

Before showing the equation of motion, consider a multi-body system of nb bodies, the composite set of generalized coordinates is $\mathbf{r} = [\mathbf{r}_1^T \ \mathbf{r}_2^T \cdots \mathbf{r}_{nb}^T]$ and $\mathbf{p} = [\mathbf{p}_1^T \ \mathbf{p}_2^T \cdots \mathbf{p}_{nb}^T]$ 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:

$$\mathbf{\Phi}(\mathbf{r}, \mathbf{p}, t) = \mathbf{0} \tag{1}$$

In addition, the Euler parameter normalization constraints must be hold:

$$\mathbf{\Phi^P} \equiv \mathbf{0} \tag{2}$$

regarding the aforementioned relationships the Newton-Euler form of constrained equations of motion in term of the Euler parameters are [5]:

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{\Phi_r}^T & \mathbf{0} \\ \mathbf{0} & 4\mathbf{G^T J' G} & \mathbf{\Phi_r}^T & \mathbf{\Phi_p}^T \\ \mathbf{\Phi_r} & \mathbf{\Phi_p} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi_p} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{r}} \\ \ddot{\mathbf{p}} \\ \lambda \\ \lambda^{\mathbf{p}} \end{bmatrix} = \begin{bmatrix} \mathbf{F}^A \\ 2\mathbf{G}^T\mathbf{n}'^T + 8\dot{\mathbf{G}}^T\mathbf{J}'\cdot\mathbf{G}\mathbf{p} \\ \gamma \\ \gamma^{\mathbf{p}} \end{bmatrix} \tag{3}$$

Where: $\mathbf{M} \equiv \text{diag}(m_1\mathbf{I}_3, m_2\mathbf{I}_3, \ldots, m_{nb}\mathbf{I}_3)$, is the mass matrix, it is a composite set of mass matrix of the nb bodies of the system; $\mathbf{J}' \equiv \text{diag}(\mathbf{J}'_1, \mathbf{J}'_2, \ldots, \mathbf{J}_{nb})')$ is the Inertia matrix; $\mathbf{F} \equiv [\mathbf{F}_1^T, \mathbf{F}_2^T, \ldots, \mathbf{F}_{nb}^T]$ and $\mathbf{n}^t \equiv [\mathbf{n}_1^{tT}, \mathbf{n}_2^{tT}, \ldots, \mathbf{n}_{nb}^{tT}]$ are the vector of external forces and torques applied on the bodies, respectively; λ and $\lambda^{\mathbf{P}}$ are the Lagrange multipliers vectors; γ and $\gamma^{\mathbf{p}}$ are the acceleration vectors, $\mathbf{G} \equiv \text{diag}(\mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_{nb})$, is the composite of $\mathbf{G}_i = [-\mathbf{e}, -\mathbf{e} + e_0\mathbf{I}_3]$; $\mathbf{\Phi_r}$ and $\mathbf{\Phi_p}$ are the Jacobian matrices of $\mathbf{\Phi}$ with respect to the position vector $\mathbf{r}$ and Euler parameters quaternion $\mathbf{p}$; $\mathbf{\Phi}^{\mathbf{p}}_{\mathbf{p}}$ and the Jacobian matrix of $\mathbf{\Phi^P}$ whit respect to quaternion $\mathbf{p}$. This system of equations, taken with the kinematic and Euler parameter normalization constraints of Eq. (1) and (2) and the associated velocity equations

$$\mathbf{\Phi_r}\dot{\mathbf{r}} + \mathbf{\Phi_p}\dot{\mathbf{p}} = -\mathbf{\Phi}_t = \nu \tag{4}$$

and

$$\mathbf{\Phi}_{\mathbf{p}}^{\mathbf{p}}\dot{\mathbf{p}} = \mathbf{0} \tag{5}$$

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 (1) is satisfied. For velocities, initial conditions should be given on $\dot{\mathbf{r}}$ and $\dot{\mathbf{p}}$.

With the equation of motion, Eq. (3), 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. Forces in contact points are considered as elastic. These forces can be derived as a collision force between two bodies that are in contact and it implies to resolve a collision detection problem.

3 Software Description

The developed software is mainly composed by three packages: the MSIM package, Collision detection package and the Haptics package. We developed the specialized functions to determine the collision between each of the geometries. These functions deliver information such as the Boolean response if the bodies are in contact, the minimal distance between the bodies, the penetration distance, the contact points and normal vector. For our purpose we considered only the following geometries: Plane, Sphere, Cylinder and Cube.

On the other hand the class Haptics functions to star and stop the servoloop, and functions to get the position of the proxy, the orientation of the pencil and the velocity vector. It is possible to set the force at the proxy.

3.1 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 Fig. 1.

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

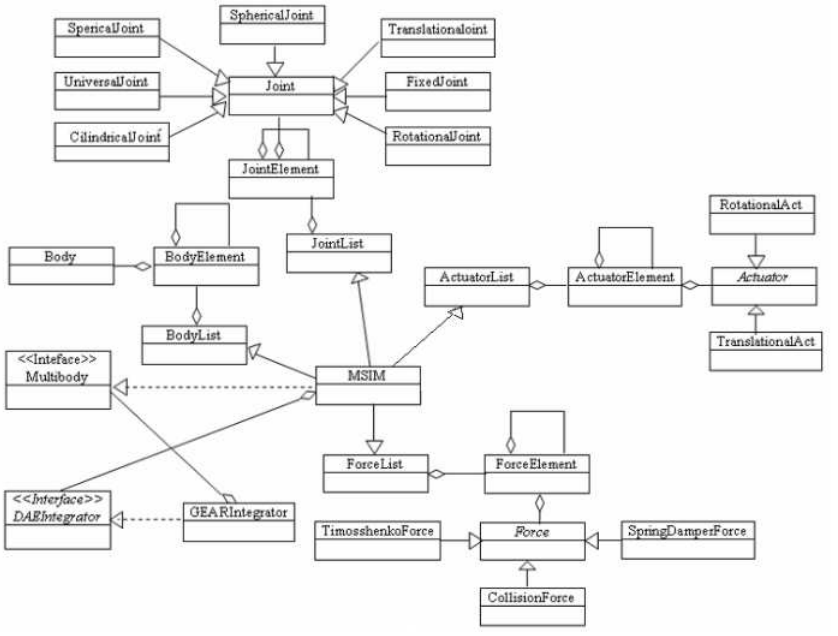

Fig. 1. UML diagram of MSIM classes

position and orientation, linear and angular velocities, mass, inertia, and forces that acts on the body (i.e., weight, coriollis force.)

Joint Class. The Joint Class is a virtual pure class which defines the behavior 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.

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.

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 specialization classes: SpringDamperForce Class, TimoshenkoForce Class and CollisionForce Class.

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 [6] integrator proposed by Gear has been implemented with some modifications. The implemented integrator considers step selection, step size, change the integrator algorithm's order. 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 LAPACK the Linear Algebra PACKage, is a software library for numerical computing written in Fortran 77.

4 Strategy for the Dynamic Model of a Hand

The strategy consists in a parallel computing architecture for model the dynamic behaviour of a hand, this is accomplished by computing the dynamics of each finger in a parallel manner. In addition, we regard other threads that execute the functions related to collision detection, haptic rendering, and graphics. We developed software to implement this idea. A UML diagram that represents the parallel computing architecture is showed in fig. 2 Each finger sends information to a Pool (a Shared Memory Segment), there is a synchronization mechanism that controls the access to the Pool. The stored information is the posture and velocity of each finger after an integration step. On the other hand, the threads read information from a Pool that contains data from kinematics and collision detection threads. We use a buffer to communicate the collision detection thread

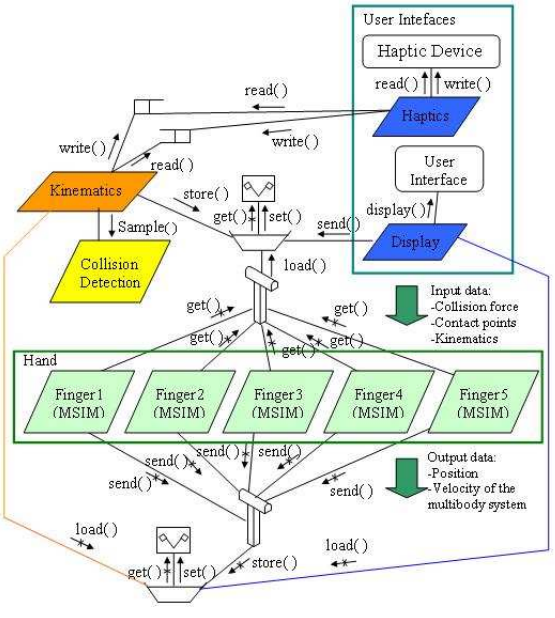

Fig. 2. Parallel computing architecture

with the haptic thread since this information exchange must be always available for a appropriate force rendering. The first buffer writes the proxy's position and orientation and the second writes the collision force for the proxy. The Kinematics thread reads information generated by the MSIM threads by means of a Pool. There is a display thread that updates the graphics presented at the User interface, this thread loads the bodies position from the Pool that stores MSIM threads output data. We consider another scenario where we include an additional virtual hand which interacts with the hand controlled by the haptic device. The virtual hand reads information from the collision detection Pool, and the simulation results are stored in an output Pool for being loaded later by the kinematics and display thread. This scenario is presented below.

5 Conclusions

A strategy for model a human hand for Haptics interaction was explained. The strategy consists in a parallel computing architecture that calculates the dynamics of a hand, this is accomplished by computing the dynamics of each finger in a parallel manner. In this approach multiple threads (e.g. haptics thread, graphics thread, collision detection thread, etc.) run concurrently and therefore we developed a synchronization mechanism for data exchange. We described the design of a C++ library for simulation of multi-body dynamics systems (MSIM). The MSIM Library is capable to compute the forward and inverse kinematic problem, and the forward and inverse dynamic problem of any multi-body system in a suitable way for haptic interaction.

References

1. Ruspini, D., Khatib, O.: A framework for multi-contact multibody dynamic simulation and haptic display. In: IEEE International Conference on Intelligent Robots and Systems, pp. 1322–1327 (2000)
2. Son, W., Kim, K., Amato, N.M., Trinkle, J.C.: A generalized framework for interactive dynamic simulation for multirigid bodies. IEEE Transactions on Systems, Man, and Cybernetics - Part B:Cybernetics 34(2), 912–924 (2004)
3. Liu, T., Yu Wang, M.: Haptic Simulation of Multibody Contact Dynamics for Fixture Loading Planning. In: Proceeding of the 2006 IEEE International Conference on Automation Science and Engineering, Shanghai, China, October 7-10 (2006)
4. Cobos, S., Ferre, M., Aracil, R., Sanchéz-Urán, M.A., Ortego, J.: Hand Gesture Recognition for Haptic Interaction. In: HCII' International 2007, 12th International Conference on Human-Computer Interaction (2007)
5. Haug, E.J.: Computer-Aided Kinematics and Dynamics of Mechanical Systems. Volume I: Basic Methods, Allyn and Bacon (1989)
6. Gear, Leimkuhler, Gupta: Automatic Integration of Euler-Lagrange Equation with Constraints. Journal of comp. and applied math. 12(13), 77–90 (1998)