Autocalibrated gravity compensation for 3DoF impedance haptic devices

A. Formaglio, M. Fei, S. Mulatto, M. de Pascale, and D. Prattichizzo

Department of Information Engineering
University of Siena, via Roma 56, 53100 Siena, Italy
{formaglio,fei,mulatto,mdepascale,
prattichizzo}@dii.unisi.it
http://www.dii.unisi.it

Abstract. The apparent mass of haptic device end-effector depends on its position inside the workspace. This paper presents a recursive algorithm to detect effective direction of gravity force, and to automatically estimate the apparent mass of the end-effector when placed at the vertices of a cubic grid contained into the device workspace. Then an on-line technique is proposed to actively compensate gravity, exploiting trilinear interpolation to compute an estimate of end-effector apparent mass in any position of the workspace. Experiments have been performed with three different haptic devices, and results shown that the apparent mass of the end-effector is compensated almost homogeneously with respect to its position in the workspace.

1 Introduction

In the last decade impedance force-feedback devices have been strongly improved thanks to both mechatronic and software developments. As a consequence, improved performance devices at lower prices are available today, as for example the Phantom Omni [3] or the Falcon [1]. Thanks to technological development, using haptic devices has become a common practice in many disciplines such as medicine, where these devices are used as training systems for laparoscopy and microsurgery [16]. Besides, haptic devices represent also an important tool to study cognitive and physiological mechanisms involved in manipulation tasks, as for example in Psychophysics or Neurophysiology [15], where the interest of experimenters is often focused on measuring finger motion, trajectories or exerted forces. In all these applications friction, inertia, and apparent mass characterizing the haptic manipulator represent a disturbance which can strongly affect experimental results.

This paper deals with the problem of gravity compensation for haptic systems. Several works can be found in the literature where different techniques are employed in order to actively cancel effects of gravity on haptic manipulators [5, 6, 9, 14]. Although the large variety of approaches, they share some common traits. Generally, gravity compensation is based on the knowledge of device kinaematics, or otherwise force sensors are used in order to achieve device mechanical transparency.

The contribution of this work consists in introducing a novel approach for gravity compensation to cancel effects of gravity force on the end-effector. Since most of

commercial haptic interfaces feature 3DoF and 3-dimensional workspace, the proposed algorithm has been designed to work with a 3DoF haptic devices. This has the main advantage of making gravity compensation independent from the particular kinaematics characterizing the device. In fact, it is well known that the mechanical properties computed at the end-effector directly depend on the particular joint configuration of the kinaematic chain [12]. For example, the *apparent mass* $M(q)$ at the end-effector, is the equivalent point-like mass due to all gravitational contributions acting on the whole manipulator in the joint configuration q. For the sake of simplicity and motivated by the fact that most of the common haptic devices have three degrees of freedom, we choose to formalize the problem of compensating the apparent mass in the 3D task space instead of the joint space. The proposed algorithm consists of two main phases:

Off-line mass estimation. A recursive procedure is applied to estimate direction of gravity, to partition the workspace using a cubic grid, and to estimate the apparent mass at each vertex of the grid. Each vertex mass is estimated using proportional-derivative (PD) position controller whose parameters depend on the mass estimated at the previous step [7]. The same PD controller is also used to move the end-effector to the next vertex of the grid.

On-line gravity compensation. Once the mass has been estimated at each vertex of the cubic grid, on-line gravity compensation is performed using trilinear interpolation to compute the apparent mass for each position internal to the grid cubes.

Validation experiments have been carried out with three haptic interfaces, with encouraging results.

The remainder of this article is structured as follows: Section 2 shows the dynamical model of the system. Section 3 decribes the off-line autocalibration, while Section 4 presents the on-line gravity compensation. Section 5 reports experiments and results. Finally, in Section 6 conclusions are drawn and future perspectives are discussed.

2 Modeling the system

2.1 End-effector point-like dynamics

The end-effector has been modeled as a point-like position-dependent mass M, subject to damping B, to gravity G and to actuators force F, according to the following non-linear dynamics:

$$M(X)\ddot{X} + B(X)\dot{X} + G(X) = F(X,\dot{X}) \tag{1}$$

where $X \in \mathbb{R}^3$ is the end-effector position, $M(X), B(X) \in \mathbb{R}^{3\times 3}$ and $F(X,\dot{X}), G(X) \in \mathbb{R}^3$. Generally, as transparency is a design requirement for impedance devices, damping factors represented by $B(X)$ are negligible, while the gravitational term $G(X)$ must be taken into account, and requires a gravity compensation control to be cancelled. Hence, equation (1) becomes:

$$M(X)\ddot{X} + G(X) \approx F(X,\dot{X}). \tag{2}$$

2.2 Feedback-linearizing controller

The algorithm proposed in [7] exploits a PD controller to hold the end-effector at the position where the apparent mass must be estimated. Hence, the force applied to the end-effector by actuators is computed as:

$$F(X,\dot{X}) = k_P(X_D - X) - k_D\dot{X} + F_G(X) \tag{3}$$

where k_P and k_D are the proportional and derivative coefficients, X_D represents the desired position to track, and $F_G(X)$ is the desired gravity compensation term.

The system represented by equation (2) has a known non-linearity in its inertial term $M(X)$, hence the PD parameters k_P and k_D can be chosen in order to linearize the closed-loop system. In the literature, this approach is referred to as *feedback linearization* [11], and allows to come up with a transformation of the open-loop system yielding a closed-loop linear system. The PD parameters can be defined as:

$$k_D(X) = \alpha M(X) \quad \text{and} \quad k_P(X) = \beta M(X) \tag{4}$$

where $\alpha,\beta \in \mathbb{R}$, and $\alpha,\beta > 0$. Equation (2), combined with the linearizing controller (4), becomes:

$$\ddot{X} = -\alpha\dot{X} + \beta(X_D - X) \tag{5}$$

which is linear and asymptotically stable. The parameters α and β can be chosen according to desired design specifications in terms of raise time T_s and steady state error which, characterize the time response to unit step inputs [4].

2.3 End-effector velocity filtering

A common issue that arises while implementing haptically enabled applications is due to sampling, and regards the noise which typically affects sampled-time signal representing end-effector velocity. After a review of most relevant works in the literature, the adaptive filtering technique introduced in [10] has been implemented. It allows to get a smooth velocity signal without increasing phase delays, as indeed it may happen using a common low-pass filter. The employed filter is referred to as discrete-time First Order Adaptive Windowing (FOAW) filter.

Common discrete-time filters typically elaborate the samples falling into a fixed time window, e.g. FIR filters. The window size should be short in order to bound time delay, and to take into account fast velocity changes. On the other hand, the window should be larger in order to produce more accurate estimates.

The FOAW filter originates from FIR filters but its time window has a variable size, changing according to a simple adaptation rule. Let $X_k \in \mathbb{R}^3$ be the end-effector position sample at the k^{th} time instant, $d = \|E_k\|_\infty$ is the peak norm of measurement noise E_k and determines the uncertainty interval for each sample, T_c is the sample time. Then the time window size n is computed as the maximum integer such that:

$$\|X_{k-n+i} - X_{k-n} - iT_c l_n\| \leq d, \quad \forall i \in \{1,2,..,n\}, \qquad l_n = \frac{X_k - X_{k-n}}{\|X_k - X_{k-n}\|}$$

where l_n is a unit vector identifying the straight line in 3D space that passes through the samples X_k and X_{k-n}. In other terms, the time window should have the maximum size such that the line identified by l_n passes through uncertainty interval of each sample falling inside the window. As the size n is set, the estimated velocity sample $\hat{V}$ is computed as:

$$\hat{V} = \frac{1}{nT_c}(X_k - X_{k-n}) \tag{6}$$

3 Off-line autocalibration

The off-line autocalibration consists of a sampled-time recursive algorithm aiming at estimating the parameters required to setup the gravity compensation. Recursion regards the apparent mass estimation $M(X_i)$ at the position X_i and PD parameters $k_P(X)$ and $k_D(X)$. In the following, initialization and recursion rules are reported, and the flow of the whole autocalibration agorithm is shown.

3.1 Initialization

A simple paradigm is applied in order to detect direction of the gravity vector, in case the device would be displaced in a different orientation from the standard one, and to achieve an intial rough estimation of end-effector apparent mass. User is asked to manually bring the end-effector almost in the center of its workspace, then a central elastic force field (with user-defined stiffness k_{init}) is activated in order to hold end-effector hanged to the position choosen by the user. At steady state, the positioning error between field center and actual end-effector position is used to detect gravity and to get an initial rough mass estimate.

Let $X_C \in \mathbb{R}^3$ be the position choosen by the user, as well as the center of the force field. Let $X(t_k) \in \mathbb{R}^3$ be the actual end-effector position at time instant t_k. The force field is defined as:

$$F(t_k) = k_{init}(X_C - X(t_k))$$

The steady state position $X(\infty)$ is considered to be reached as soon as n_{init} consecutive samples fall within a spherical interval of radius ε_{init}, where ε_{init} is a sufficiently small parameter defined by the user. Hence, the unit vector representing the desider direction of gravity compensation is computed as:

$$v_G = -\frac{X(\infty) - X_C}{\|X(\infty) - X_C\|} \tag{7}$$

while an initial rough mass estimation is computed as:

$$\hat{M}_0 = \frac{k\|X_C - X(\infty)\|}{g} \tag{8}$$

where $g = 9.81\frac{m}{s^2}$. As already mentioned, recursion applies also to PD parameters, which depend on end-effector apparent mass in order to feedback-linearize the system. Hence, PD parameters are initialized as:

$$k_D(0,0) = \alpha\hat{M}_0 \quad \text{and} \quad k_P(0,0) = \beta\hat{M}_0 \tag{9}$$

The last initialization procedure consists in creating the cubic grid which discretizes the device workspace. The cube side length L is a user-defined parameter. For the sake of simplicity, henceforth we will assume that the algorithm works in a cubic subspace entirely contained inside the device original workspace, leaving the extension to the whole workspace to future developments. Hence, the cubic grid is created starting from the origin of the device reference frame, $X_0 = (0,0,0)$, then the other grid vertices are defined as (nL, mL, pL), where n, m and p are integers belonging to the interval $[-N, N]$, being N such that the entire grid is contained within the device workspace.

3.2 Mass estimation recursion rules

In order to estimate the apparent mass at each vertex X_i, an iterative technique introduced by De Luca and Panzieri in [7] is used. In what follows, we briefly report the basic idea. Let X_i be the i^{th} vertex where the mass must be estimated, while X is the current position, available from haptic interface encoders. A PD control scheme is used to bring end-effector towards the desired vertex position X_i. The steady state tracking error between X_i and the actual position X is used to get an estimation of the gravity force. This procedure is iterated to obtain a mass estimation $\hat{M}(X_i)$ at the i^{th} vertex. Hence, at the j^{th} iteration step, the force to render at the end-effector is computed as:

$$F_j(X, \hat{V}) = k_P(i,j)(X_i - X) - k_D(i,j)\hat{V} + M_j(X_i) g v_G. \tag{10}$$

where $\hat{V}$ is the filtered signal representing the end-effector velocity. The term $M_j(X_i) g v_G$ represents the gravity compensation contribution computed at the j^{th} iteration, aligned with the direction v_G, previuosly detected. The apparent mass estimation is updated at steady state, according to the following recursion rule:

$$M_{j+1}(X_i) = \frac{k_P(i,j)\|X_i - X\|}{g} + M_j(X_i), \tag{11}$$

where $M_0(X_i)$ is initialized to the mass estimated for the closest previous vertex:

$$M_0(X_i) = \hat{M}(X_{i-1}). \tag{12}$$

The i^{th} vertex mass estimation $\hat{M}(X_i)$ is achieved as soon as n_M consecutive samples fall within a spherical interval of radius ε_M, where ε_M is a sufficiently small user-defined parameter. The procedure is then iterated for the next vertex as well.

While estimating apparent mass at the i^{th} vertex, PD parameters used at the j^{th} iteration are based on $(j-1)^{th}$ mass estimation. Hence, the following recursion rule is used:

$$k_D(i,j) = \alpha M_{j-1}(X_i) \quad \text{and} \quad k_P(i,j) = \beta M_{j-1}(X_i) \tag{13}$$

On the other hand, while the PD controller is used to move the end-effector towards the next vertex, PD parameters are computed relying on mass estimation at the previous vertex, and such values hold as initialization for the mass estimation in the next vertex. Hence:

$$k_D(i,0) = \alpha \hat{M}(X_{i-1}) \quad \text{and} \quad k_P(i,0) = \beta \hat{M}(X_{i-1}) \tag{14}$$

The off-line autocalibration is summarized in Algorithm 1.

Algorithm 1 Off-line autocalibration

initialize
  detect gravity direction v_G (Eq. 7)
  initialize mass $\hat{M}_0$ (Eq. 8)
  initialize PD parameters $k_P(0,0), k_D(0,0)$ (Eq. 9)
for each vertex i **do**
  goto position X_i (Eq. 10)
  repeat
    update mass $M_j(X_i)$ (Eq. 10 and 11)
    update PD $k_P(i,j), k_D(i,j)$ (Eq. 13)
  until n_M consecutive samples within a sphere of radius ε_M
  initialize mass for next vertex (Eq. 12)
  initialize PD for next vertex (Eq. 14)
end for

4 On-line gravity compensation

On-line gravity compensation is performed while using the impedance device in virtual reality applications. It consists of applying a force contribution in addition to haptic rendering implemented in the end-user application. Let Z_E be the virtual environment impedance, the total force to apply to the end-effector is:

$$F(X) = Z_E(X) + F_G(X) \tag{15}$$

where X is the end-effector position, while $F_G(X)$ is the gravity compensation term, and relies on data acquired during the off-line autocalibration. Since apparent mass estima-

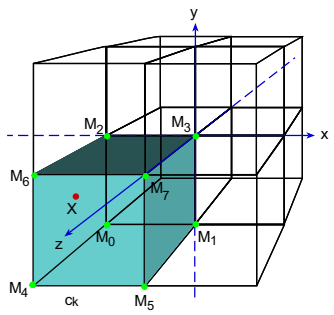


Fig. 1. Trilinear interpolation: given the end-effector position X (red point), the cube c_k is selected, and estimation of apparent masses at its vertices are used to compensate gravity.

tions are avaliable only at the vertices of the cubic grid, the value of apparent mass in an arbitrary position X of the workspace is computed by detecting which is the cube of the grid containing the end-effector, and using trilinear interpolation between the mass values of its vertices (see Figure 1). Let us suppose that $X = (x,y,z)$ is contained inside the

cube c_k, and let $X_{k_0} = (x_k, y_k, z_k)$ be the vertex of cube c_k having minimum coordinates. Interpolation coefficients and vertex masses are computed as:

Coefficients:

$$\begin{cases} b = (x - x_k)/L \\ a = (L - b)/L \\ d = (y - y_k)/L \\ c = (L - d)/L \\ f = (z - z_k)/L \\ e = (L - f)/L \end{cases}$$

Apparent masses at vertices of cube c_k:

$$\begin{cases} M_0 = M(x_k, y_k, z_k) \\ M_1 = M(x_k + L, y_k, z_k) \\ M_2 = M(x_k, y_k + L, z_k) \\ M_3 = M(x_k + L, y_k + L, z_k) \\ M_4 = M(x_k, y_k, z_k + L) \\ M_5 = M(x_k + L, y_k, z_k + L) \\ M_6 = M(x_k, y_k + L, z_k + L) \\ M_7 = M(x_k + L, y_k + L, z_k + L) \end{cases}$$

Finally, the apparent mass used to compensate the gravity in position X is computed as:

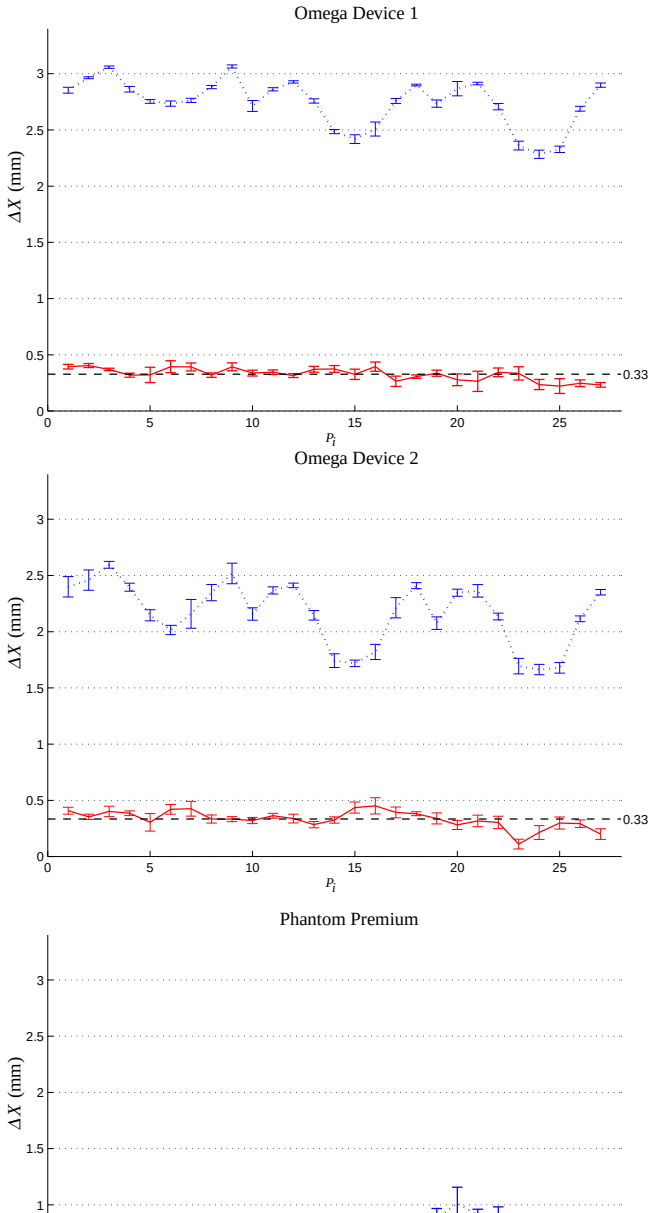
$$M(X) = ecaM_0 + ecbM_1 + edaM_2 + edbM_3 + fcaM_4 + fcbM_5 + fdaM_6 + fdbM_7$$

5 Experiments

Experiments were performed using three different haptic devices with 3DoF: two, nominally identical Omegas [2] and one PHANToM Premium [13]. First, off-line autocalibration was run on these devices to estimate the end-effector apparent mass in a cubic grid featuring 27 vertices, where each internal cube had $L = 40$mm side length. The user-defined parameters where $k_{init} = 0.5\frac{N}{mm}$, $n_{init} = n_M = 30$ and $\varepsilon_{init} = \varepsilon_M = 0.1mm$. All devices were placed in standard orientation.

A probe mass $m_P = 30$g was attached at the end effector of each device, and a virtual contact between the probe mass and a horizontal surface was simulated in 27 different points P_n inside the cubic grid, with no external actions by the user. The contact points were defined as $P_n = (30i,\ 30j,\ 30k)$mm, where $i, j, k \in \{-1,\ 0,\ 1\}$. Virtual contact was rendered using elastic impedance local model, with stiffness $k = 1\frac{N}{mm}$. Experimental data consist of the steady-state values of virtual penetration $\Delta X = \|P_n - X\|$ (also referred to as *penalty*), recorded both with and without on-line gravity compensation, 10 times for each P_n and for each haptic device.

In Figures 2 we reported experimental results obtained with the 1^{st} Omega, the 2^{nd} Omega and the PHANToM Premium. Each figure reports the mean penalties ΔX (averaged over 10 repetitions) and the corresponding standard deviations, for each contact point, with and without gravity compensation. In absence of gravity compensation (blue dotted plots), the end-effector apparent mass $M(X)$ at position X adds to the probe mass m_P, and affects the dynamics of virtual contact point. Infact, as expected, the steady state values ΔX measured by all devices without gravity compensation may remarkably change depending on position of contact point, since the apparent mass of the end-effector depends on its position in the task space. On the other hand, when gravity compensation is running (red solid plots), it can be seen that the dependency of ΔX on the position of the contact point is remarkably reduced, if not almost cancelled for all devices.



P_i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
x_i	0	0	30	30	30	0	-30	-30	-30	0	0	30	30	30	0	-30	-30	-30	0	0	30	30	30	0	-30	-30	-30
y_i	-30	-30	-30	-30	-30	-30	-30	-30	-30	0	0	0	0	0	0	0	0	0	30	30	30	30	30	30	30	30	30
z_i	0	30	30	0	-30	-30	-30	0	30	0	30	30	0	-30	-30	-30	0	30	0	30	30	0	-30	-30	-30	0	30

Fig. 2. Mean penalty ΔX (averaged over 10 repetitions) and related standard deviation, over contact points, with (red solid line) and without (blue dotted line) gravity compensation. The table on the bottom reports the coordinates of each contact point.

A further result stems from experiments. If the gravity compensation was exact, the expected penalty in any position of the workspace would be $\overline{\Delta X} = \frac{m_P\, g}{k} \approx 0.3mm$. As represented by the black dashed line in Figures 2, the mean penalty ΔX_M, averaged over all repetitions and all point of contacts, obtained with on-line gravity compensation, is equal to 0.33mm for both Omegas and to 0.35mm for the PHANToM, that are very close to expected value 0.3mm computed above. In other terms, ΔX_M depends almost on the probe mass m_P, since the apparent mass $M(X)$ is cancelled by gravity compensation.

6 Conclusions and future works

This paper presents a technique for compensating effects of gravity for a 3DoF impedance haptic device. The proposed algorithm consists of two phases. The first is an off-line recursive automatic calibration, in which the effective direction of gravity force is detected, and apparent mass of the end-effector is measured at the vertices of a cubic grid contained inside the device workspace. The second phase is the on-line gravity compensation, and is performed exploiting trilinear interpolation to compute an estimate of end-effector apparent mass in any position inside the cubes of the grid. Experiments performed with three devices, shown that the proposed algorithm is able to get reliable measures of apparent mass at the grid vertices, and to compensate effects of gravity force. As a result, the apparent mass of the end-effector is almost cancelled, homogeneously with respect to the position in the workspace.

By comparing blue dotted plots of Figure 2 corresponding to both Omegas with the one of the PHANToM, it stems that, in absence of gravity compensation, Omegas and PHANToM devices exhibit quite different behaviors. If this could be expected, what is indeed worth remarking is that different behaviors can be observed even between both Omegas, although they are nominally identical (see blue dotted plots obtained with both Omegas). This may depend on several factors such as device age, mechanical couplings, joints friction, which in turn can affect the apparent mass of the end-effector. The proposed technique is based on measurements performed on the actual device of interest, hence it can take into account also such phaenomena and consequently can cancel their effects on end-effector behavior.

In summary, the advantages of the proposed techinque are manifold: it is based on measurements performed on the actual device of interest; it is independent from device kinaematics; it is able to detect the actual direction of gravity force. On the other hand, the main drawback of this approach is that in case of end-effector replacement or device orientation change, the off-line calibration must be performed again in order to estimate new gravitational properties of the system in the new configuration. Moreover, the preliminary algorithm proposed in this paper only works in a cubic subspace of the device workspace, whose dimensions are supposed to be known a-priori. Future developments of the algorithm will make it working in the whole task space, even in absence of information about shape and dimensions of device workspace.

The final target of this work consits of integrating the proposed algorithm in the Haptik Library [8], in order to make it transparently available for virtual reality developers.

References

1. Website: home.novint.com/products/novint_falcon.php.
2. Website: www.forcedimension.com.
3. Website: www.sensable.com/haptic-phantom-omni.htm.
4. S.P. Boyd and C.H. Barratt. *Linear Controller Design: Limits of Performance*. Prentice Hall, 1991.
5. M.C. Çavusoglu, D. Feygin, and F. Tendick. A critical study of the mechanical and electrical properties of the PHANToM TM haptic interface and improvements for high performance control. *Presence*, 11(6):1–18, 2002.
6. D. Checcacci, E. Sotgiu, A. Frisoli, CA Avizzano, and M. Bergamasco. Gravity compensation algorithms for parallel haptic interface. pages 140–145, 2002.
7. A. De Luca and S. Panzieri. An Iterative Scheme for Learning Gravity Compensation in Flexible Robot Arms. *Automatica*, 30(6):993–1002, 1994.
8. M. de Pascale and D. Prattichizzo. The Haptik Library: a component based architecture for uniform access to haptic devices. *IEEE Robotics and Automation Magazine*, 2007. In press.
9. JT Feddema and JL Novak. Whole arm obstacle avoidance for teleoperated robots. pages 3303–3309, 1994.
10. F. Janabi-Sharifi, V. Hayward, and C.S.J. Chen. Discrete-time adaptive windowing for velocity estimation. *Control Systems Technology, IEEE Transactions on*, 8(6):1003–1009, 2000.
11. H.K. Khalil. *Nonlinear systems*. Prentice Hall Upper Saddle River, NJ, 2002.
12. O. Khatib. Inertial Properties in Robotic Manipulation: An Object-Level Framework. *The International Journal of Robotics Research*, 14(1):19, 1995.
13. T.H. Massie and J.K. Salisbury. The phantom haptic interface: A device for probing virtual objects. volume 55, pages 295–300, 1994.
14. R. Ott, M. Gutierrez, D. Thalmann, and F. Vexo. Improving user comfort in haptic virtual environments trough gravity compensation. 2005.
15. D. Prattichizzo and S. Rossi. Special Issue on Robotics and Neuroscience. *Brain Research Bullettin*, 2007. In Press.
16. RM Satava and SB Jones. Current and future applications of virtual reality for medicine. volume 86, pages 484–489, 1998.