INTERACTION TECHNIQUE FOR VIRTUAL ROBOT STABILIZATION WITH DYNAMIC BEHAVIOR

Técnica de interacción para la estabilización de robots virtuales con comportamiento dinámico

Ulises Zaldívar C.[1], Diego Murillo C.[1], Xiomara P. Zaldívar C.[1], Edson F. Osuna P.[1], José V. Núñez N.[2]
zaldivar@uas.uasnet.mx / murillo@uas.uasnet.mx / xiomara@uas.uasnet.mx / edson@uas.uasnet.mx / pepenalda@hotmail.com

Recibido: 12 marzo, 2009 / Aceptado: 28 octubre, 2009 / Publicado: 31 diciembre, 2009

ABSTRACT. Virtual robots used in virtual environments are normally modeled with kinematic behavior. The use of dynamic behavior in virtual environments allows the simulation of more realistic virtual worlds. In this paper we present an interaction technique to manipulate a robot in virtual environment considering the physically based modeling. The virtual robot is constituted by two models. The first model has a kinematic behavior and it's called the *Kinematic Virtual Robot* (KVR). The second one has the dynamic behavior and it's called the *Dynamic Virtual Robot* (DVR). The robot modeled in this research is based on the architecture of the Scara Unimate S-103 manipulator, with 3.5 degrees of freedom. The interaction technique used is based on the application of spring-dampers connecting the KVR joints with the DVR joints. In this research we made a comparative study of the application of torsional spring-dampers and linear spring-dampers in order to establish the convenience of the spring-damper for the robot stabilization during its manipulation and during the tasks execution. We observed that the technique based on linear spring-damper keeps more stabilized the virtual robot than the torsional spring-damper technique. The virtual robot manipulation is performed by the user with a Phantom Omni™ haptic device.

KEYWORDS: Virtual robot, dynamic behavior, spring-damper model, dynamic stabilization.

RESUMEN. Los robots virtuales utilizados en ambientes virtuales tienen generalmente comportamiento cinemático. La integración del comportamiento dinámico en ambientes virtuales permite simular mundos virtuales más realistas. En este trabajo se presenta una técnica de interacción para manipular un robot en un ambiente virtual considerando el modelado basado en física. El robot virtual está compuesto de dos modelos. El primero con comportamiento cinemático, llamado Kinematic Virtual Robot (KVR) y el segundo con comportamiento dinámico, llamado Dynamic Virtual Robot (DVR). El modelo del robot tomado como base es el manipulador Scara Unimate S-103, con 3.5 grados de libertad. La técnica de interacción utilizada se basa en la aplicación de resortes-amortiguadores que conectan las articulaciones del KVR con las del DVR. Se presenta un estudio comparativo de la aplicación de resortes-amortiguadores torsionales y lineales, con el fin de determinar el más adecuado para lograr la estabilización del robot virtual durante su manipulación y durante la ejecución de una tarea. Se concluyó que la técnica basada en resortes lineares mantiene más estabilizado al robot que la basada en resortes torsionales. La manipulación del robot es realizada por medio de la interfaz Phantom OMNI™.

Palabras clave: Robot virtual, comportamiento dinámico, modelo resorte-amortiguador, estabilización dinámica

[1] Facultad de Informática Mazatlán – Universidad Autónoma de Sinaloa (UAS), Av. De los Deportes y Av. Universidad SN, Mazatlán, 82000, Sinaloa, México
-info.maz.uasnet.mx
[2] Instituto Tecnológico de la Laguna, Blvd. Revolución y Calzada Cuauhtémoc, Torreón, Coahuila, México
-www.itlalaguna.edu.mx

Introduction

Virtual robotics laboratories are used in order to support research or undergraduate and graduate programs in the robotics area as well as to realize robots teleoperation by computer simulation [1-7]. Most of the virtual robotics laboratories simulate virtual robots (VR) with a kinematic behavior, where the robots perform offline or online movements by user manipulation [5, 7-8]. This kinematic behavior doesn't allow a realistic simulation in a virtual environment (VE) [9], so it is not recommended for dynamic virtual worlds, where objects may exist with a dynamic behavior. Virtual worlds with dynamic behavior of objects allow the creation of very realistic virtual worlds, and robots in VE must be improved with this behavior in order to reproduce the dynamic movements of real robots. As a way to obtain realistic dynamic behavior, while the contact between virtual objects is taking place, a spring-damper model was adopted based on using an artificial coupling between a tracker device and the dynamic virtual object. This technique [10] has been used in other research works [11-16]. Borst and Indugula [12-13] applied the technique of a spring-damper model to a virtual hand in order to produce virtual realistic grasping. Our approach is mainly based on the spring-damper model [14-16], but focuses on the manipulation of a VR in order to avoid its penetration with other objects in the scene. Each joint of the VR is represented in the VE by two 3D models coupled by a spring-damper. These models are called *Kinematic Virtual Robot* (KVR) and *Dynamic Virtual Robot* (DVR) (analogous to the *tracked part* and the *visual part* [14-15]). During the manipulation of the VR, the KVR follows the DVR's position and orientation. The DVR is the rendered 3D model of the robot in the virtual scene. It mimics real robot dynamic behavior during its operation. The KVR is an off-screen rendered model used to compute the springs.

System architecture

The hardware used for this research is constituted by a PC-based system with AMD Athlon™ Dual Core processor, 2.29 GHz, 768 MB of RAM memory and ATI Radeon 3100 Graphic Card, 384 MB of video RAM. For user interaction, the Phantom Omni™ haptic interface is used. Software architecture is made by C++ programming language, using OpenGL libraries for visualization, AGEIA PhysX for the dynamic behavior and collision detection, OpenHaptics libraries for controlling the haptic interface and Solidworks™ 2001 CAD system for robot joints and objects design (Figure 1).

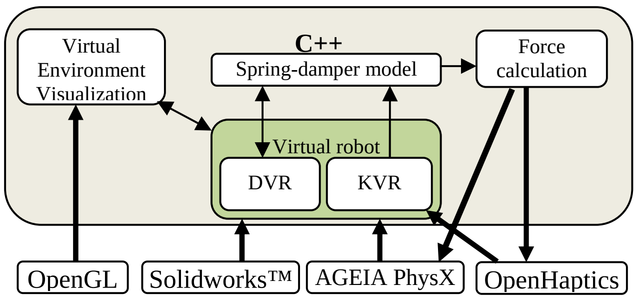

Figure 1. Software architecture

The virtual environment is developed in C++, where several libraries are integrated. OpenGL is used for 3D geometrical objects visualization, and the use of textures enhances the visual realism for the scene. Virtual robot design is made in Solidworks™ CAD system, and the files are converted to polygonal meshes imported from the virtual environment. Each polygonal mesh is used to create the corresponding objects for the DVR and the KVR. PhysX software receives the polygonal meshes and creates both models (the DVR and the KVR) of the VR. PhysX characteristics allow us to create the kinematic behavior and the dynamic behavior. OpenHaptics is used to manipulate the KVR and receives the calculated force which is sent to the haptic device.

Manipulation of the Kinematic Virtual Robot

The Phantom Omni™ haptic interface allows the manipulation of the VR by the application of geometrical transformations to the KVR. The configuration of this device is constituted by 3 joints with 6 degrees of freedom (DOF). As the Unimate S-103 manipulator is constituted by 3.5 DOF, we need to make a correlation with some articular coordinates from the Phantom to the virtual Unimate. In Figure 2 and Figure 3 are represented the kinematic diagrams for the Unimate manipulator and the Phantom device respectively. The relation between both is established by linking the values of θ_1, θ_2 and θ_3 in the Unimate manipulator with the configuration of θ_1, θ_4 and θ_6 in the Phantom Omni™ respectively.

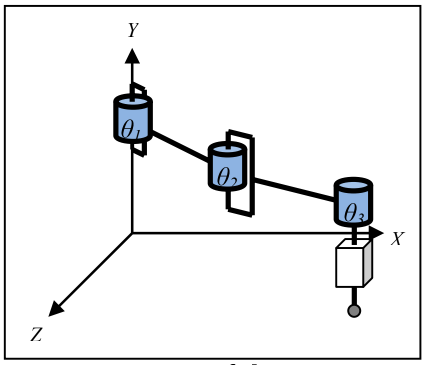

Figure 2. Kinematics of the Unimate S-103

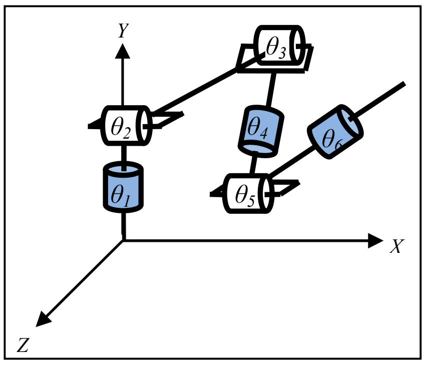

Figure 3. Kinematics of the Phantom Omni™

Once the coordinate values of the Phantom Omni™ (θ_1, θ_4 and θ_6) are read, they are passed to the values of the KVR joints orientation (θ_1, θ_2 and θ_3). Then the localization for each KVR joint is obtained considering the length and height of each joint. The next equations present the calculation of the geometrical transformation for each joint:

${}^{W}M_{J1} = \begin{bmatrix} R_{J1} & \vec{T}_{J1} \\ \vec{0} & 1 \end{bmatrix}$	${}^{W}M_{J2} = \begin{bmatrix} R_{J2} & \vec{T}_{J2} \\ \vec{0} & 1 \end{bmatrix}$	${}^{W}M_{J3} = \begin{bmatrix} R_{J3} & \vec{T}_{J3} \\ \vec{0} & 1 \end{bmatrix}$
$R_{J1} = R\left(Ang(\theta_1), Axis(0,1,0)\right)$	$R_{J2} = R\left(Ang(\theta_2), Axis(0,1,0)\right)$	$R_{J3} = R\left(Ang(\theta_3), Axis(0,1,0)\right)$
$\vec{T}_{J1} = Pos\left(0,10,0\right)$	$\vec{T}_{J2} = R_{J1} \times Pos\left(10,14.6,0\right) + \vec{T}_{J1}$	$\vec{T}_{J3} = R_{J2} \times Pos\left(14,12.65,0\right) + \vec{T}_{J1} + \vec{T}_{J2} + \vec{T}_{G}$

${}^{W}M_{J1}$ is the transformation matrix for Joint 1 in the KVR. This matrix is composed by the rotation matrix R_{J1} and the translation vector $\vec{T}_{J1}$. R_{J1} is obtained by the angle-axis rotation $R\left(Ang(\theta_1), Axis(0,1,0)\right)$ using θ_1 as angle. $\vec{T}_{J1}$ is obtained by the initial position of Joint 1. The vector $Pos\left(0,10,0\right)$ represents the initial translation of the joint. Following the same formulation we obtain the transformation matrix for Joint 2, with some changes in the calculation of the translation vector $\vec{T}_{J2}$. It is obtained with the transformation of the relative position, with respect to Joint 1, $Pos(10,14.6,0)$ multiplied by the Rotation of Joint 1. Finally we must to add the translation of the vector $\vec{T}_{J1}$. In the calculation of the transformation matrix ${}^{W}M_{J3}$ of Joint 3, we must consider the translation of the two previous joints, the relative position of Joint 3, and a possible offset if the gripper is in movement.

Manipulation by spring-damper model

In order to perform the dynamic behavior, we control the KVR by the use of the OpenHaptics library, which provides functions to communicate the C++ language with the Phantom device. When the user moves the Phantom articulations, a callback function in the system reads the values and calculates the geometrical transformation for each joint of the KVR. During this manipulation phase, the virtual robot must have a dynamic behavior that represents the robot movement in the real world. In order to provide the user with this sensation of realism, we adopt the approach of coupling the DVR and the KVR by a spring-damper. This technique is inspired from the use of virtual springs to couple a tracked part to a visual dynamic part in order to perform realistic virtual assembly with physically based simulation [14-16].

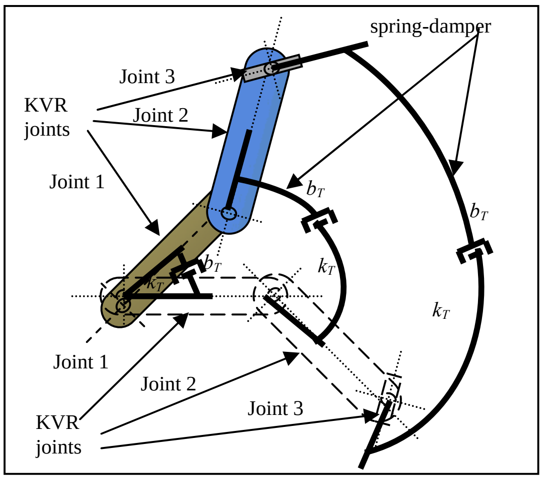

Figure 4. Torsional spring-dampers

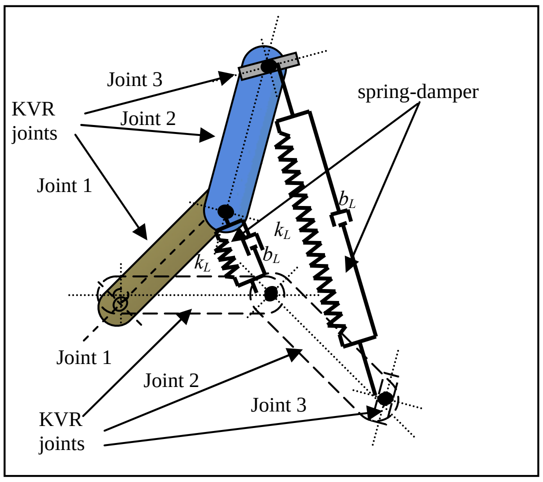

Figure 5. Linear spring-dampers

By coupling the DVR and the KVR with spring-damper, we allow realistic behavior for the virtual robot by avoiding 3D models to pass through each other. Contact force rendering is computed using the difference of positions and orientations of the KVR and the DVR. When the user moves the virtual robot, the DVR (visual model) tends to follow the KVR (tracked model). Collision detection and response prevent the visual model to penetrate into other objects in the scene. As the DOF of Joint 1 and Joint 2 in the Unimate S-103 manipulator have cylindrical movement, we considered, in first instance, to use two torsional spring-dampers to perform this dynamical-cylindrical movement (Figure 4). We observed, with this implementation, some instability in the virtual robot (presented in the Results section), then we considered the implementation of two linear spring-dampers (Figure 5), instead of the torsionals, to eliminate this instability. The values of spring constant and damping constant were obtained by performing manipulation trials of the virtual robot. These values were adjusted until the most stable movement of the robot was obtained.

Torsional spring-damper

The torsional spring-damper produces a torque (rotational force) that is applied to the correspondent joint.

First, we calculate the rotational difference between Joint 1 in the DVR and the KVR ($\theta_{D,J1\to K,J1}$):

$$ {}^{W}R_{D,J1\to K,J1}{}^{t} = {}^{W}R_{D,J1}^{-1}{}^{t} \times {}^{W}R_{K,J1}{}^{t}, \qquad \theta_{D,J1\to K,J1} = Ang\left({}^{W}R_{D,J1\to K,J1}{}^{t}\right) $$

${}^{W}R_{D,J1}^{-1}{}^{t}$ is the inverse of the rotation matrix of Joint 1 in the DVR at time *t*, ${}^{W}R_{K,J1}{}^{t}$ is the rotation matrix of Joint 1 in the KVR at time *t*, ${}^{W}R_{D,J1\to K,J1}{}^{t}$ is the rotation matrix which represents the rotation of ${}^{W}R_{D,J1}{}^{t}$ with respect to ${}^{W}R_{K,J1}{}^{t}$ and $\theta_{D,J1\to K,J1}$ is the angle extracted from the matrix ${}^{W}R_{D,J1\to K,J1}{}^{t}$.

Then we obtain the restoring torque RT with the multiplication of the torque constant k_T and the angle $\theta_{D,J1\to K,J1}$: $RT = k_T \times \theta_{D,J1\to K,J1}$. This torque is applied around the axis $Axis_{D,J1\to K,J1}$, extracted from ${}^{W}R_{D,J1\to K,J1}{}^{t}$: $Axis_{D,J1\to K,J1} = Axis\left({}^{W}R_{D,J1\to K,J1}{}^{t}\right)$

Then we must compute the damping torque separately, because the torque is applied in the opposite direction of the restoring torque. For the damping, it is necessary to compute the angular velocity for each joint in both models (the DVR and the KVR). The angular velocity of Joint 1 in the DVR ($R_{Dw,J1}$) is obtained by: $R_{Dw,J1} = R\left(Ang\left(\frac{Ang\left({}^{W}R_{D,J1^{t}->D,J1^{t-1}}\right)}{\Delta t} \right), Axis\left({}^{W}R_{D,J1^{t}->D,J1^{t-1}}\right) \right)$

where ${}^{W}R_{D,J1^{t}->D,J1^{t-1}}$ is the rotation matrix that represents the existent rotation of Joint 1 in the DVR at time *t* with respect to its previous rotation (its rotation at time *t-1*), obtained by: ${}^{W}R_{D,J1^{t}->D,J1^{t-1}} = {}^{W}R_{D,J1}^{-1}{}^{t} \times {}^{W}R_{D,J1}{}^{t-1}$

Δt is the time interval between *t-1* and *t* situations.

The same formulation is used to obtain the angular velocity of Joint 1 in the KVR ($R_{Kw,J1}$).

Then we obtain the relative velocity $\theta_{Kw,J1\to Dw,J1}$ of Joint 1 in the KVR with respect to Joint 1 in the DVR:

$$ \theta_{Kw,J1\to Dw,J1} = Ang\left({}^{W}R_{Dw,J1} \times {}^{W}R_{Kw,J1}^{-1}\right) $$

And the magnitude of the damping torque is obtained by $DT = b_T \times \theta_{Kw,J1\to Dw,J1}$, where b_T is the damping constant. The damping torque is applied around the axis $Axis_{K,J1^{t}->K,J1^{t-1}}$ extracted from ${}^{W}R_{K,J1^{t}->K,J1^{t-1}}$:

$$ Axis_{K,J1^{t}->K,J1^{t-1}} = Axis\left({}^{W}R_{K,J1^{t}->K,J1^{t-1}}\right) $$

The same method is used to calculate the spring-dampers in Joint 2 and Joint 3 of the virtual robot.

Linear spring-damper

The linear spring-damper produces a force (linear force) that is applied to the correspondent joint. For the linear spring-damper, we introduced an attached one mass for each joint (in the KVR and in the DVR) (Figure 5). The masses in the KVR (KVR masses) are transformed cinematically, whereas the masses in the DVR (DVR masses) have a dynamic behavior. When the KVR model changes its orientation, the KVR masses change theirs positions. These positions are used to calculate the linear spring-dampers that attach the DVR with the KVR. For this calculation, we must know the positions of each mass in the KVR and in the DVR. The positions for the masses in the KVR are represented by ${}^{W}T_{K,J1}$ for the mass in Joint 1, and ${}^{W}T_{K,J2}$ for the mass in Joint 2. The positions for the masses in the DVR are represented by ${}^{W}T_{D,J1}$ for the mass in Joint 1, and ${}^{W}T_{D,J2}$ for the mass in Joint 2. The linear spring-damper for Joint 1 is calculated by: $\vec{F} = k_L\left({}^{W}\vec{T}_{K,J1}{}^{t} - {}^{W}\vec{T}_{D,J1}{}^{t}\right) - b_L\left(\vec{T}_{Dw,J1} - \vec{T}_{Kw,J1}\right)$.

${}^{W}\vec{T}_{K,J1}{}^{t}$ and ${}^{W}\vec{T}_{D,J1}{}^{t}$ are the positions of the KVR mass and the DVR mass at time *t*, respectively. $\vec{T}_{Dw,J1}$ is the linear velocity for the DVR mass and $\vec{T}_{Kw,J1}$ is the linear velocity for the KVR mass. k_L and b_L are the spring and damper constants respectively. The linear velocity for the masses is calculated with the difference of current positions for each mass with their precedent position, divided by the time interval Δt:

$$\vec{T}_{Dw,J1} = \frac{\left({}^{W}\vec{T}_{D,J1}{}^{t} - {}^{W}\vec{T}_{D,J1}{}^{t-1}\right)}{\Delta t} \text{ and } \vec{T}_{Tw,J1} = \frac{\left({}^{W}\vec{T}_{T,J1}{}^{t} - {}^{W}\vec{T}_{T,J1}{}^{t-1}\right)}{\Delta t}$$

Similar formulation is used to calculate the spring-damper for Joint 2.

The force $\vec{F}$ is applied to the mass in the DVR, and this mass follows the position of the KVR mass. As the DVR masses are linked to the DVR, the joints of this robot tend to follow the KVR with rotational movements. Some cylindrical constraints assure the position of the axis for each joint.

Experiments

For the torsional approach, we manipulated the Phantom device with the hand, performing random movements, and recorded the angular value for the KVR and the DVR in each time instant. The expected behavior was that the DVR followed the KVR as close as possible. We made a comparison between the angular values of the KVR with the DVR on Joint 1 and Joint 2. Then we programmed automatic movements for both joints in the KVR, and recorded the same values (of KVR and DVR). In this last case, we established preset values for the joints. Joint 1 moves from 90 degrees to -90 degrees, and Joint 2 moves from -45 degrees to 45 degrees. We made an interpolation of these values in an interval of 1 second, and then interpolate back the values. We repeated immediately the first interpolation process. The angles for both joints where interpolated in the same time. For the linear approach we made the same experiments realized for the torsional approach. In **Table 1** are presented the experiments configurations.

Table 1. Realized experiments

	Exp 1 (E1)	*Exp 2 (E2)*	*Exp 3 (E3)*	*Exp 4 (E4)*
Spring-damper type	Torsional	Linear	Torsional	Linear
Manipulation type	By user	By user	Automatic	Automatic

Results and discussion

In order to evaluate the virtual robot stabilization for the torsional spring-damper and the linear spring-damper, the data collected from the experiments 1 and 2 were compared, as well as the data collected from the experiments 3 and 4. The results show a significant improvement with the use of the linear spring-damper compared with the use of the torsional spring-damper in the two cases: user manipulation and automatic movement (Table 2).

In E1, the results show a difference in the orientation of the joints principally when their direction is changed (Figure 6.a). The maximum angle error for Joint 1 is 10.2°, and for Joint 2 is 26.3°. The average of errors is 1.6° and 4.1° for Joint 1 and Joint 2 respectively. The results collected in E2 show a minimal difference for the orientation of the DVR joints compared with the KVR joints. Just when direction is changing, there is a small angular error (Figure 6.b). The maximum angle error for Joint 1, in E2, is 1.7°, and for Joint 2 is 4.0°. The average of errors is 0.5° for both joints.

Table 2. Angular difference between the KVR joints and the DVR joints

	E1	*E2*	*E3*	*E4*
Max. angular difference (Joint 1)	10.2	1.7	16.6	3.4
Max. angular difference (Joint 2)	26.3	4.0	23.5	6.0
Average angular difference (Joint 1)	1.6	0.5	8.6	0.3
Average angular difference (Joint 2)	4.1	0.5	5.2	0.6

In E3, when the virtual robot is performing the automatic movement (without user interaction), the results show a difference in the orientation of the joints, principally for Joint 1. Joint 2 presents principally some angular offset of the DVR with respect to the KVR (Figure 7.c). The maximum angle error for Joint 1 is 16.6°, and for Joint 2 is 23.5°. The average of errors is 8.6° and 5.2° for Joint 1 and Joint 2 respectively.

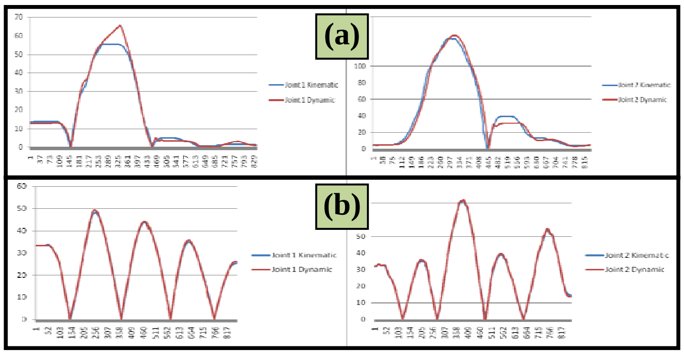

Figure 6. Results for E1 (a) and E2 (b)

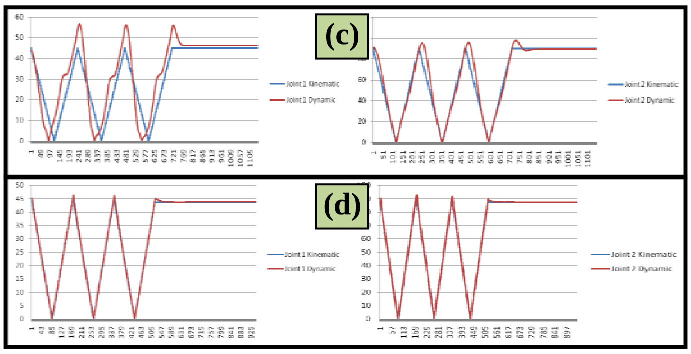

Figure 7. Results for E1 (c) and E2 (d)

The results collected in E4 show a minimal angular error. We observed that the lines representing the DVR angles and the KVR angles are almost overlapped in comparisons for Joint 1 and comparisons for Joint 2. Just when direction is changing, there is a small angular error (Figure 7.d). The maximum angle error for Joint 1, is 3.4°, and for Joint 2 is 4.0°. The average of errors is 0.34° for Joint 1, and 0.6° for Joint 2.

Conclusions and future work

This paper presented a spring-damper model to achieve dynamic behavior of a virtual robot. We made a comparative study for two types of springs: the torsional one and the linear one. Experimental results showed that the use of the linear springs instead of the torsional ones, in this approach, keeps the dynamic virtual robot more stabilized. For the linear spring, we attach virtual masses in particular coordinates of the robot joints. This work presented only a stabilization study for the VR, but different experiments may be interesting to carry out, as the user performance evaluation and/or communication evaluation in teleoperation (compared with VR where its behavior is kinematic). The user interface used for the manual manipulation is the Phantom Omni™ which supports haptic feedback, but in this approach has been used only to track 3 of its joints.

The immediate future work for this approach will be the calculation of the contact force between the robot and other objects in the scene, and the application of this force in the haptic device for user contact sensation. The next future approach is the implementation of this spring-damper technique in more complex virtual robots (for example in PUMA architecture).

References

1. Ibarra Zannatha J. M., Zaldívar Colado U., Wiederhold Grauert P. (2001). An Approach to Internet Robotics: Generation of Interactive Virtual Worlds and Internet Teleoperation. *1st International Conference on Information Technology in Mechatronics (ITM'01). 108-115.*
2. Ibarra Zannatha J. M., Zaldívar Colado U., (2001). Desarrollo de un Sistema de Teleoperación y Programación Automática para un Laboratorio Virtual de Robótica. *Tercer Congreso Mexicano de Robótica (AMROB'01)*
3. Ibarra Zannatha J. M., Zaldívar Colado U., Iturbe Córdoba E. J., López Trujillo J., Montano Gella L., Mínguez J. (2002). 3D Mapping for Mobile Robots Using Interactive Virtual Worlds and Internet Teleoperation. *Proceedings of the 15th IFAC World Congress of the International Federation of Automatic Control (IFAC'02).*
4. Torres Medina F., Candelas Herías F. A., Puente Méndez S. T., Ortiz Zamora F. G., Pomares Baeza J., Gil Vázquez P. (2002). Laboratorio virtual remoto para la enseñanza de robótica. *III Jornadas de trabajo Enseñanza vía Internet-Web de la ingeniería de sistemas y automática (EIWISA'02).* Alicante. 65-69.
5. Zaldívar Colado U. Tesis de Maestría. (2003).Robótica Asistida por Teleoperación y Realidad Virtual. Centro de Investigación y de Estudios Avanzados – IPN, México. http://www.cs.cinvestav.mx/Estudiantes/TesisGraduados/2003/tesisUlisesZ.pdf
6. Candelas Herías F. A., Torres Medina F., Gil Vázquez P., Ortiz Zamora F. G., Puente Méndez S. T., Pomares Baeza J. (2004). Laboratorio virtual remoto para robótica y evaluación de su impacto en la docencia. *RIAI: Revista Iberoamericana de Automática e Informática Industrial*. **1**(2): 49-57.
7. Zaldívar Colado X. P., Zaldívar Colado U., Carvajal Valdés R., Niebla Zatarain J. C. (2008). Diseño de un laboratorio virtual de robótica como apoyo a la docencia en la Educación Superior. *XXIV Simposio Internacional de Computación en la Educación (SOMECE 2008).*
8. Rohrmeier, M. (2000). Web based robot simulation using VRML. *Proceedings of 2000 Winter Simulation Conference.* **2**: 1525 – 1528.
9. Zaldívar Colado U., Garbaya S. (2009). Virtual Assembly Environment Modelling. *ASME AFM World Conference on Innovative Virtual Reality (ASME AFM WINVR 09).*
10. Colgate J. E., Grafing P. E., Stanley M. C., Schenkel G. (1993). Implementation of stiff virtual walls in force-reflecting interfaces. *Proceedings IEEE virtual reality annual international symposium (VRAIS)*. 202–208.
11. Adams R. J., Hannaford B. (1998). A two-port framework for the design of unconditionally stable haptic interfaces. *Proceedings IROS'98*.
12. Borst C. W., Indugula P. (2005). Realistic virtual grasping. *Proceedings IEEE virtual reality conference (VR'05)*. 91–98, 320.
13. Borst C. W., Indugula P. (2006). A spring model for whole-hand virtual grasping. *Presence Teleoperators Virtual Environ*. **15**(1): 47–61.
14. Zaldívar Colado U., Garbaya S., Blazevic P. (2006). Spring-damper Model for Parts Mating in Virtual Assembly Environment. *International Symposium on Robotics and Automation (IEEE-ISRA 2006)*. 587-593.
15. Garbaya S., Zaldívar-Colado U. (2007). The affect of contact force sensations on user performance in virtual assembly tasks. *Virtual Reality: Springer London.* **11**(4): 287-299.
16. Garbaya S., Zaldívar-Colado U. (2007). Modelling Dynamic Behaviour of Parts in Virtual Assembly Environment. *ASME AFM World Conference on Innovative Virtual Reality (ASME AFM WINVR 09).*