

A Generic Force-Server for Haptic Devices

Lorenzo Flückiger^a and Laurent Nguyen^b

^aNASA Ames Research Center, Moffett Field, CA

^bRecom Technologies, Moffett Field, CA

ABSTRACT

This paper presents a novel architecture allowing a generic force-feedback device to be used by different software tools dedicated to teleoperation and mission planning. This architecture relies on a “force-server” program running between the real-time controller of the haptic device and a set of applications using it. Possible applications include mission ground control system interfaces, telemetry systems coming back from real robots, or external simulation programs.

The force-server concept is based on a high-level description of the forces to be generated. This description consists of spatial constraints defined by their type (point, line, plane and mesh), position and orientation. A force profile (space and/or time dependent) is assigned to each constraint. This description enables the generation of complex fields of forces by combining basic constraints.

The advantage of this method is that an application can send force updates by simply modifying several parameters of the constraint. Between two updates (from the application) the force-server is able to continuously compute new forces corresponding to the actual position of the device handle. This approach enables the control loop of the force feedback device to easily run at 500Hz when the application may send updates only at 25Hz.

This novel method widens the use of force-feedback devices by providing a common interface to the different applications, and allowing multiple clients to use the same haptic device. A testbed using a 6 DOF haptic device has been developed. The device generates forces coming simultaneously from three different sources: a Java interface to experiment various force profiles, a rover simulator, and a scientific visualization tool used during planetary missions.

Keywords: Force Feedback, Haptic Device, Virtual Reality, Force Server, Robot Manipulator/Rover Simulation

1. INTRODUCTION

Human-Machine interfaces exploiting multiple modalities of interaction are generally more efficient and intuitive than classical user interfaces. Visual displays are widely used in this context and are often coupled with haptic displays, especially in the domain of teleoperation. Unlike visual displays for which a huge number of software and hardware techniques are available, haptic display solutions are still scarce or tailored to very specific applications.¹⁻⁵

In this context, the Microengineering Department of the Swiss Federal Institute of Technology, Lausanne (EPFL) has developed, since 1998, a new concept of input device with force-feedback. This device – called *FIDEL* (Force Interaction DELta) – has been designed to interact with robotic systems through Virtual Reality (VR) based interfaces.

This haptic device has been integrated with the teleoperation facilities developed by the Autonomous and Robotics Area (ARA) of the NASA Ames Research Center (Moffett Field, CA). Since the ARA already uses distributed visual displays for mission planing and science analysis, a novel concept of *Force-Server* has been developed to handle the haptic device in a similar way as the other tools: the device becomes a haptic display driven by different applications.

This paper first presents the haptic device used for this research as well as the others software tools related to the project. Then, the next part describes the concept of Force-Server and its implementation. Finally, the last part of the paper highlights the results already achieved trough some examples of applications.

2. PLATFORM

The testbed developed at the ARA consists of the haptic device FIDEL with its controller connected to several software clients.

Correspondence: E-mail: lorenzo@artemis.arc.nasa.gov

2.1. Haptic device: FIDEL

FIDEL, shown on Figure 1, is a 6 degrees of freedom (DOF) input device with force-feedback capabilities. It has been developed at EPFL.^{6,7} The key point of the structure is to decouple translations and rotations:

- The translational structure is based on the DELTA⁸ parallel robot structure: three kinematic chains passively constrain a moving platform to be parallel to a fixed reference.
- Rotations are achieved by a “wrist” module mounted on the moving platform: it could passively act like a joystick or be actively driven by three small motors.

This specific design provides the haptic device with a high stiffness, large workspace and low inertia thanks to the parallel structure and the main actuators mounted on the fixed base. The working volume for translations is comprised in a cylinder of 30cm of diameter by 20cm of height. The wrist allows rotations of $\pm 20^\circ$ in all the three axes. Translational forces can be as high as 25N in the main part of the working volume.

FIDEL efficiently exploits the human “arm+hand” capabilities: the user can easily translate his hand in the 3D space, and the joystick like rotational part takes in account the limited range of rotation of the human wrist. This haptic device was primarily designed to control robots through a virtual reality based interface, but its usage has already been widened to applications in the medical or microscopy fields.

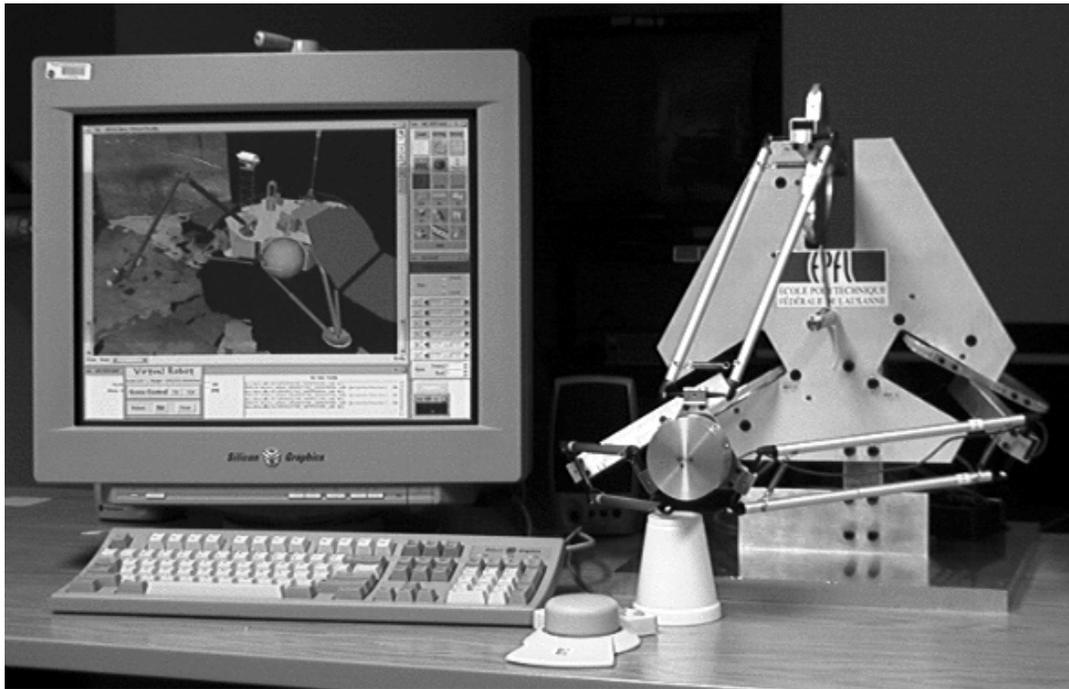


Figure 1. The testbed (real-time controller and amplifier not shown). The force-feedback input device on the right allows the control of the virtual Robotic Arm (fixed on the Mars Polar Lander) represented on the computer screen on the left.

2.2. Software clients

The FIDEL haptic device is designed to enhance the use of virtual reality based interface. To test it in real situations it has been interfaced with the virtual environments currently used. For this research, two VR applications have been combined to form a complete testbed: VIZ (developed by the ARA) for data visualization and science analysis and VirtualRobot (mainly developed at EPFL) for robot simulation and control.

2.2.1. VIZ: Virtual Reality based interface

VIZ⁹ is the virtual reality based interface currently used by the ARA to support missions like the Mars Polar Lander. This interface is designed to be a 3D graphics rendering program running as a server: any client application can send data and commands to VIZ. As a result, complex interfaces, involving multiple programs which all share the same final graphic rendering, can be built rapidly. The current VIZ interface allows clients to display realistic environments in real-time and provides scientists with several efficient measurement tools (among many other capabilities).

2.2.2. VirtualRobot: robots simulation

VirtualRobot⁶ is a program which automatically solves the kinematic behavior of any robot manipulator, and provides a virtual environment to interact with these virtual robots. This allows the user to study and control robot manipulators in a very efficient and intuitive manner. VirtualRobot has been extended to simulate multi-wheeled robots driving on uneven surfaces. Thus, the program can be used for fixed manipulator like the Robotic Arm on the Mars Polar Lander (see Figure 1) as well as for any kind of rover.

2.3. Communication

A major issue occurring with force-feedback devices is the update frequency of the graphics workstation which is more than an order of magnitude lower than the update frequency of the haptic device. The VR interface on the workstation typically gives a refresh rate about 18 to 75Hz (mainly limited by the graphics rendering) which is sufficient for a “smooth” rendering. On the other hand, the controller of the force-feedback device has to control each motor at a rate of at least 500Hz (1kHz is the current frequency used) to avoid jerky behavior. Several approaches are suitable to address this problem. One common approach consists of making the real-time controller interpolate forces at 1kHz between the reception of new inputs from the graphic workstation. The Force-Server concept is a more advanced method to overcome this issue.

3. FORCE-SERVER

The haptic device must know at any time the current force it has to generate – *output-force* – corresponding to the current position of the handle, the world to simulate, and any other information varying over time that the application needs to “render” as forces. Existing approaches to compute output forces use either a physical representation of the world (the output-force is computed regarding a given set of equations), or directly a given discrete force-field (for example the force-field can be computed from an elevation map.^{10–17} The physical representation is well-suited for a haptic rendering of wide range of physical phenomenon, but the need to code the equations could be a real drawback when changing the world or adding/removing other properties is needed. In addition, this representation is not adequate to render other information not linked to a physical phenomenon, like a repulsive field generated by a hazardous area. The force-field method is completely general and even allows the description of environments with discontinuities. The drawback of the force-field description is the high volume of information needed for an accurate representation (specifically near singularities like a wall into which a rover should not collide) and the difficulty to generate the desired force-field in a simple way (even if the force-field is generated by a set of physical equations).

The next sections present the proposed novel approach to compute the output-forces, removing drawbacks, and in a more general framework allowing multiple applications to work together.

3.1. Concept

The goal of the Force-Server is to provide a general interface for haptic devices. The main idea is to create a Force-Server program running between the real-time controller and any application using the haptic device as shown in Figure 2. The Force-Server program is responsible for computing the output-force resulting from the request of all the connected clients. Possible applications includes a virtual reality based interface, a real robot sending data corresponding to its surrounding environment, or a program generating forces regarding changing parameters of any physical simulation. All these applications are able to send information about forces to the Force-Server in a standardized form. The format of the force description allows the real-time controller to get local information (in space and time) on how to compute forces. The Force-Server receives asynchronous updates from clients and generates continuous output forces for the controller.

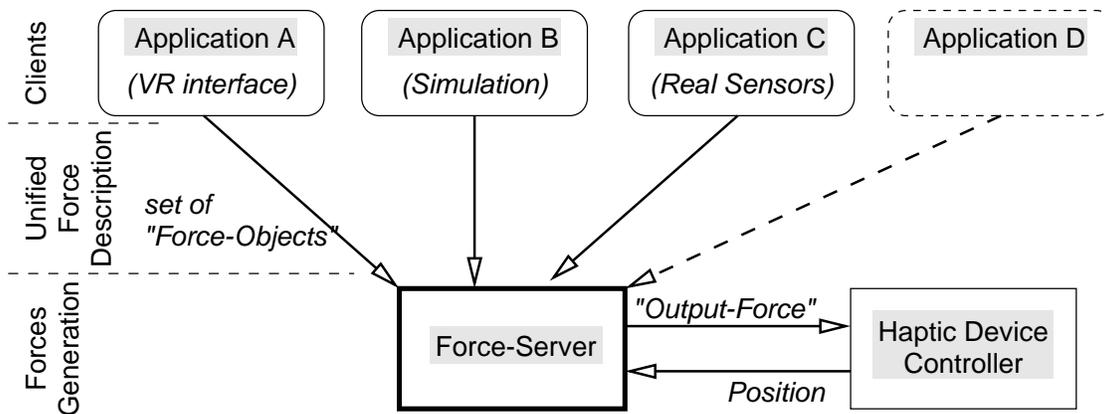


Figure 2. Concept of the Force-Server

3.2. Force Description

The force description used by the Force-Server is the key point of this new concept. Basically, the world is modeled by a set of high level force descriptions. A force description is called a *force-object* and corresponds in general to a physical object of the world (e.g., a rock), or to any other type of data (e.g., battery power remaining to go somewhere). Each force-object is composed by a *force-primitive* and a *force-profile*. The force-primitive defines the distribution of force vector directions. The force-profile defines the intensity of the force to be generated. Regarding the current position of the haptic device handle in the space with respect to the force-object, the direction and intensity of the force generated by this object is computed (see Figure 3 for some examples of force-objects). Then each individual force is combined to obtain the output-force that the haptic device should generate. The current method computes the final force using a linear addition of all the force vectors.

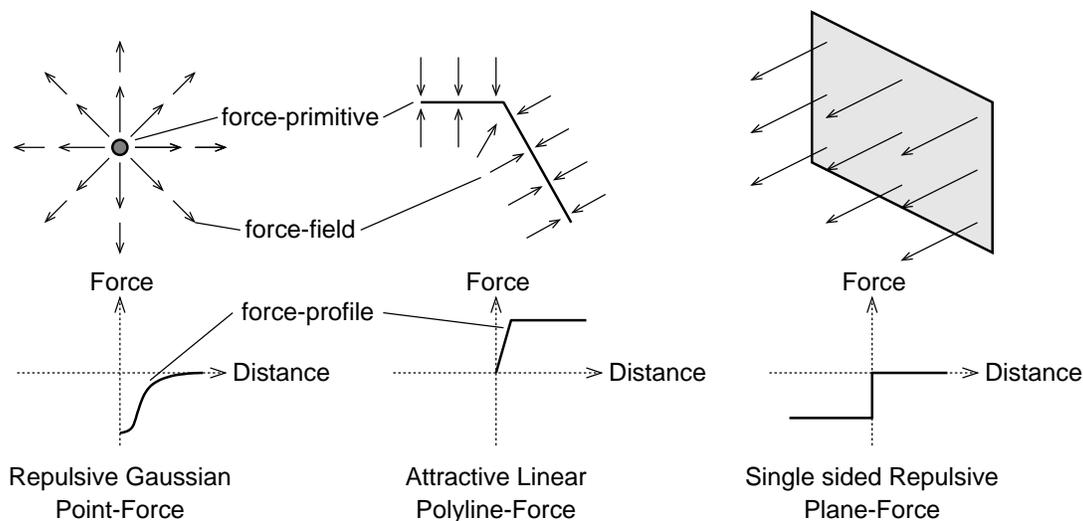


Figure 3. Examples of force-field generated by several force-primitives combined with various force-profiles

To offer a description of all the force fields encountered, a minimal set of four force-primitives have been defined. These four force-primitives are described in Table 1.

Currently, seven different force-profiles are provided by the interface, and custom profiles can also be defined. Moreover, each profile can have the global characteristic of being attractive or repulsive (see Figure 3 for examples).

The different profiles described in Table 2 are based on simple classical continuous functions. They have been chosen for their strong characteristics or their useful usage in common situations. It should be noted that each function representing a

<i>Primitive</i>	<i>max. DOFs remaining</i>	<i>Usage example</i>
Point	0	Repulsive dangerous objects / Attractive targets
PolyLine (set of lines segment)	1	Move along a constrained path
Plane	2	Simulating walls (repulsive), constrain a rover to move on a floor (attractive)
Surface (3D polygonal mesh)	3	General complex object representation / shape understanding

Table 1. Force-primitives available for defining forces/constraints

force-profile has an adjustable threshold: each associated force-object could only generate a limited force (which could be the maximum output force of the device or any lower value to minimize the maximum effect of a force object compared to the others). Figure 4 shows an example of a force-profile with a threshold limiting the force near the origin (right image).

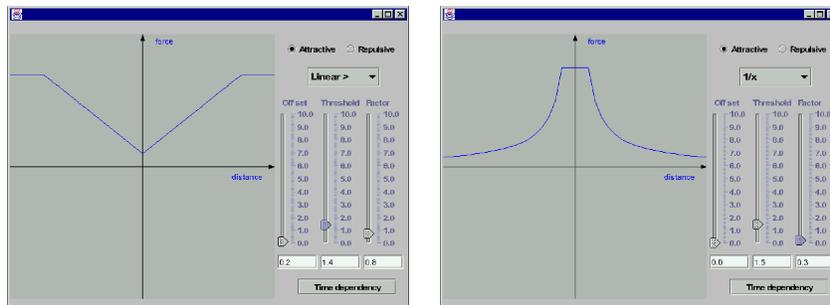


Figure 4. Example of two force-profiles with force limits.

Currently, each force-profile is entirely defined by three components. Thus, a minimal amount of information is required to describe a complete force-object:

- Name of the force object (unique identifier)
- Type of force-primitive
- Type of force-profile
- Three components for profile definition (for Plane and Mesh primitive, three additional components are needed to describe the other side of the primitive)

As a result, clients of the Force-Server describe forces with compact messages. This feature is critical for clients frequently updating the force-objects needed to represent their force environment. For example, a simulation of a rover driving will send to the Force-Server force-objects limiting the haptic device to move only in certain directions, in order to reflect the non-holonomic behavior of the rover. In this case, the simulation will need to update the force-objects at the frequency of the simulation (approx. 25Hz).

The combination of the proposed force-primitives with force-profiles allows to virtually describe any kind of force in space. The forces described by a set of force-objects could be seen as a force-field: each point in space is subject to a specific force calculated with respect to the current state of all the force-objects. The advantage over existing methods, like potential-field description, is a higher level of force definition as well as a real flexibility for the user. This force-object description can also be seen as constraints in the 3D space. For instance, a *plane* force-primitive coupled with a strongly attractive force on both sides of the plane will constrain the handle to move only in this given plane. Table 1 gives the number of degrees of freedom (DOF) that the different force-primitives allow if the force-object is considered as a constraint.

Profile	Description	Usage example
Constant	force constant	Unified linear force field
Linear Increasing	$f = a * dist + offset, a > 0$	Simple constraint to remain close to a feature
Linear Decreasing	$f = a * dist + offset, a < 0$	Regular (linear) attraction/repulsion near a feature
Gaussian	Gaussian with peek for $dist = 0$	Attractive point target
Quadratic	$f = a * dist^2 + offset$	Strong constraint to stay on a feature
Inverse	$f = a / dist + offset$	Strong attraction/repulsion near a feature
Custom	custom function	Function optimized for a specific application

dist=distance from the current location to the force primitive

Table 2. Force-profiles available for defining forces/constraints

With the proposed method, several clients can send their force requests to the haptic device. The Figure 5 illustrates with a simple example the concept of the Force-Server listening to two clients. The Force-Server continuously listens to the updates coming from these clients, and computes the new output-force that the haptic device should generate.

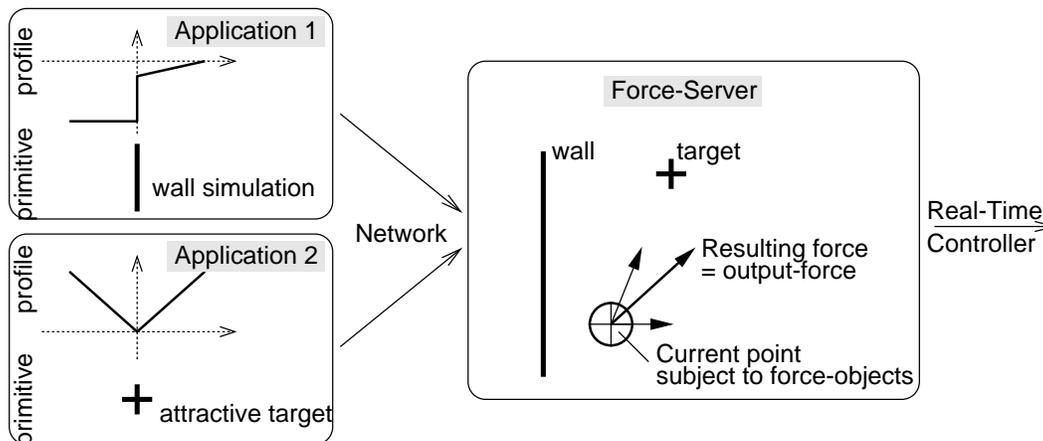


Figure 5. Simple example of the Force Server concept

4. RESULTS

The current implementation of the Force-Server is a Java-based program. It consists of several modules:

- The communication module which listens to any client applications requesting the sending of forces commands to the device.
- The central module manages a database of all the different force descriptions received from the clients, and computes a resulting force by combining the different primitives.
- A force editor provides the user with a convenient interface to define new forces and edit their respective parameters.
- A 3D visualization module allows the user to visualize the state of the haptic device, the different forces, as well as the resulting force*.

*The program can also send this information to the ARA visualization tool VIZ, which permits a faster representation including all the objects composing a 3D world.

A realistic demonstration has been built with three clients. The goal was to drive a rover on a uneven surface (in a virtual environment). The first client is the force-editor: it constrains the haptic device to come back at its initial position (point-constraint) as if the handle was linked with a spring to a virtual center point. The second client is the rover simulator which constrains the movement of the haptic device only in the direction current of the rover. This force-primitive is a plane with its normal aligned with the wheels axes. The third client is the visualization software – VIZ –in which the user has defined hazardous areas (closed polylines constraints).

4.1. Evaluation of the Force-Server

The actual implementation of the Force-Server has demonstrated that the concept is completely functional. The different experiments performed have shown how straightforward it is to define new force environments and the flexibility of the description. In particular, the usage of the Force-Editor (shown on Figure 6) is a great help to generate and test new behaviors defined by different force environments. The Force-Editor can be used “on-line” and allows the change of all the parameters dynamically. Finally, a set of force-objects can also be saved to disk and reloaded to resume an experiment.

However, the communication delay between the Force-Server and the real-time controller might introduce vibrations under certain conditions. This result was expected^{18,19}: the closed-loop system composed by the human hand and the haptic device (for example the hand pushes the handle and force-feedback pushes back the hand) becomes unstable when delays longer than 10ms are introduced.

This delay between the Force-Server and the haptic device is due to the current version of the controller: a standalone PC running a real-time OS (RealLink) which controls different input/output boards and listens to the network through a UDP channel. The real-time OS offers extremely rapid control capabilities but poor advanced programming solutions. Consequently, the prototype of Force-Server was running on a separate computer. The new version of the controller uses a standard PC running WindowsNT. This will allow the force server to run on the same machine. The Java version has demonstrated the potential of the method, therefore a C++ version will be implemented to offer optimal performances.

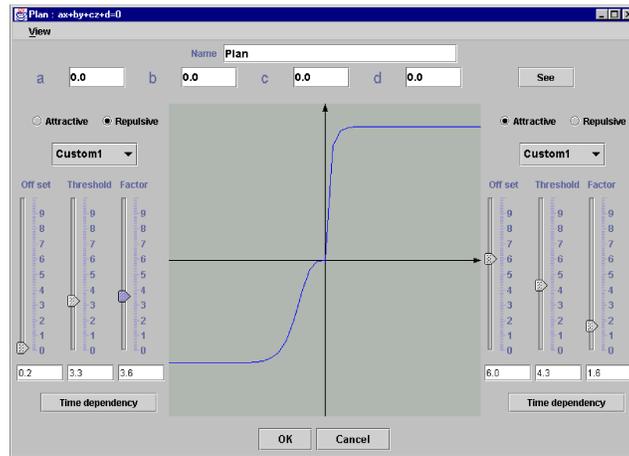


Figure 6. The Force-Editor window.

4.2. Future developments

The current implementation of the Force-Server with the haptic device uses the 6 DOF provided by the device for position and orientation control. However, the actuators generating torques for the rotations are not activated on this testbed. The Force-Server does not generate output torques, either. Even if the model could easily be extended with force-primitive object generating torques, such behavior will not find real usage in classical teleoperation application: it will be more interesting to generate torques by applying multiples constraints at different location on a solid, resulting in an output force plus an output torque.

Another major issue, only partially solved in this research, is to keep a coherence between the displacements/forces generated on the haptic device and the movements of the objects in the virtual world. A classical example of a moving point in the 3D space over a rugged surface does not present any problem. However, when the haptic device is used for more advanced control,

the direct relation between the displacement of the haptic device handle and the associated object in the 3D virtual space could disappear. For example, imagine that the haptic device is used to control a robotic arm, and the goal is to feel a given surface with the end effector. It could be possible to move the handle in a position not reachable by the manipulator (out of workspace). In this case, it becomes more difficult to decide what kind of force should be applied on the handle because it is not supposed to be here! The usage of a “proxy” – a point remaining in the robot envelope or on the surface, linked to the handle and exercising a return spring force – is used in our implementation to solve this issue. However, more complex cases involving multiples client applications lead inevitably to a “slide” between the world of the haptic device and the simulated synthetic world.

5. CONCLUSION

A novel architecture for the control of haptic devices has been proposed and a complete testbed has demonstrated the validity of the concept. The Force-Server allows multiple clients to render forces through a common interface. The unified force description based on *force-primitives* and *force-profiles* is efficient and can represent almost all types of force fields. A force editor provides the user with a convenient and easily understandable interface to build new force environments. Examples of applications involving robot manipulators/rovers simulations and virtual reality based interfaces showcase the easiness to build complex environments with heterogeneous applications. For these reasons, we strongly believe that the Force-Server will be a valuable concept for both the developers of haptic devices and the end-users of such devices.

CREDITS

This work was funded by the Swiss National Science Foundation and supported by the NASA Ames Research Center (CA).

We wish to thank Michael Frossard for its helpful contribution to this research and for his implementation of the Java Force-Server.

REFERENCES

1. G. Burdea, *Force and touch feedback for virtual reality*, John Wiley & Sons, New York, 1996.
2. H. Iwata, “Artificial reality with force-feedback: development of a desktop virtual space with compact master manipulator,” *Computer Graphics* **24**(3), pp. 165–170, 1990.
3. T. H. Massie and J. K. Salisbury, “The PHANTOM haptic interface: A device for probing virtual objects,” in *Proceedings of the 1994 ASME International Mechanical Engineering Congress and Exhibition*, vol. DSC 55-1, pp. 295–302, (Chicago), 1994.
4. P. J. Berkelman, R. L. Hollis, and S. E. Salcudean, “Interacting with virtual environments using a magnetic levitation haptic interface,” in *International Conference on Intelligent Robots and Systems (IROS)*, pp. 117–122, (Pittsburgh, Pennsylvania), 1995.
5. Y. Tsumaki, H. Naruse, D. N. Nenchev, and M. Uchiyama, “Design of a compact 6-dof haptic device,” in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2580–2585, (Leuven, Belgium), 1998.
6. L. Flückiger, *Interface pour le pilotage et l’analyse des robots basée sur un générateur de cinématiques générales*. Ph.d. thesis 1897, Swiss Federal Institute of Technology, Lausanne (EPFL), 1998.
7. L. Flückiger, C. Baur, and R. Clavel, “CINEGEN: a rapid prototyping tool for robot manipulators,” in *The Fourth International Conference on Motion and Vibration Control (MOVIC’98)*, G. Schweitzer, R. Siegwart, and P. Cattin, eds., vol. 1, pp. 129–134, (Zürich, Switzerland), 1998.
8. R. Clavel, *Conception d’un robot parallèle rapide à quatre degrés de liberté*. Ph.d. thesis 925, Swiss Federal Institute of Technology, Lausanne (EPFL), 1991.
9. L. A. Nguyen, M. Bualat, L. J. Edwards, L. Flückiger, C. Neveu, K. Schwer, M. Wagner, and E. Zbinden, “Virtual reality interfaces for visualization and control of remote vehicles,” in *ICRA 2000, Proceedings of the IEEE International Conference on Robotics and Automation 2000*, IEEE, (San Francisco (CA)), April 2000.
10. H. Iwata, “Pen-based haptic virtual environment,” in *IEEE Annual Virtual Reality International Symposium*, pp. 287–292, IEEE Service Center, (Seattle, WA, USA), 1993.
11. H. Iwata and H. Noma, “Volume haptization,” in *IEEE Proceedings of the Symposium on Research Frontiers in Virtual Environments*, pp. 16–23, 1993.
12. K. Hirota and M. Hirose, “Providing force feedback in virtual environments,” *IEEE Computer Graphics and Applications* **15**(5), 1995.

13. C. B. Zilles and J. Salisbury, "A constraint-based god-object method for haptic display," in *Human Robot Interaction and Cooperative Robots*, IEEE, ed., vol. 3, pp. 146–151, IEEE, 1995.
14. R. S. Avilla and L. M. Sobierajski, "A haptic interaction method for volume visualization," in *IEEE Proc. Visualization '96*, pp. 197–204, 1996.
15. N. Ahuja and J.-H. Chuang, "Shape representation using a generalized potential field model," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**, February 1997.
16. D. C. Ruspini, K. Kolarov, and O. Khatib, "The haptic display of complex graphical environments," in *Computer Graphics Proceedings, Annual Conference Series*, 1997.
17. J. P. Fritz and K. E. Barner, "Design of haptic data visualization system for people with visual impairments," *IEEE Transactions on rehabilitation engineering* **7**(3), pp. 372–384, 1999.
18. F. Vahora, B. Temkin, T. M. Krummel, and P. J. Gorman, "Development of real-time virtual reality haptic applications: Real-time issues," in *Proceedings of the 12th IEEE Symposium on Computer-Based Medical Systems*, IEEE, ed., pp. 290–295, 1999.
19. R. J. Adams and B. Hannaford, "Stable haptic interaction with virtual environment," *IEEE Transactions on Robotics and Automation* **15**, pp. 465–474, June 1999.