Workshop in Virtual Reality Interactions and Physical Simulation VRIPHYS (2008)
F. Faure, M. Teschner (Editors)

A Desktop Virtual Reality System with Physical Animation and Glove Interaction

J. Aleotti[1] and S. Caselli[1]

[1]Robotics and Intelligent Machines Laboratory, Dipartimento di Ingegneria dell'Informazione, University of Parma, Italy

Abstract

This paper describes the on-going development of a desktop virtual reality system which offers real-time user interaction and realistic physics-based animation of rigid objects. The system is built upon a graphical engine which supports scene graphs, and a physics-based engine which enables collision detection. Full hand pose estimation is achieved through a dataglove and motion tracker. Motion of the user's hand is coupled to a 3D model of the human hand. A virtual grasping algorithm ensures stability and allows manipulation tasks to be performed even on complex shapes such as triangle meshes. The system supports ballistic motion of falling objects and models grasped objects with a spring-damper scheme. Moreover, vibratory output is generated as feedback to the user.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Virtual reality, I.3.5 [Computer Graphics]: Physically based modeling

1. Introduction

The global computer and video game industry revenue has become larger than the movie industry income. Interactive entertainment applications are rapidly growing in all technical aspects and are driving factors behind architectural innovation. Such applications push the boundaries of computing to achieve more realistic virtual experiences. The next frontier towards realism is the development of robust physics systems that enable on-the-fly content creation. There are two classical methods for achieving real-time simulation of physical behavior. The first technique is constraint-based modelling, which uses assembly constraints related to the geometric properties of objects. The second method is physics-based modelling, which uses Newtonian physics to describe the motion of objects. In general, the physics-based modeling approach is computationally expensive as it requires the computation of the motion of virtual bodies by numerically integrating the equations of motion.

Having a natural mechanism for interaction within a virtual environment is another key point in the development of advanced 3D applications. The use of natural user interfaces greatly improves the feeling of immersion and the degree of realism. The most advanced interfaces for natural human interaction in virtual reality involve the use of devices such as datagloves, force feedback actuators and helmets. Datagloves enable gesture interaction and object manipulation which provide strong enhancement as compared with traditional tangible devices. Extensive work has been conducted for rigid-body simulation techniques as well as for collision detection approaches and object manipulation. However, combinations of physically-based modeling, efficient collision detection and object manipulation with user feedback are reported with lower frequency.

This paper describes the on-going development of a desktop virtual reality system which is oriented to object manipulation and assembly. The virtual environment enables real-time user interaction and physics-based animation of rigid objects. The system is built upon a graphical engine which supports scene graphs. Hierarchical models can be imported in the virtual environment. Complex shapes described by triangle meshes can also be simulated. Besides the graphical infrastructure the system integrates a physics-based engine which enables fast and accurate collision detection. The user can interact with the virtual objects wearing a dataglove and motion tracker. Motion of the user's hand is coupled to a 3D model of the human hand. A virtual grasping algorithm has also been developed. The algorithm evaluates the stability of a grasped object according to a standard grasp quality met-

ric. Once a grasp is recognized as stable the virtual object is linked to the virtual hand and it can be carried in the simulated environment. Currently the system does not allow full dextrous manipulation of grasped objects. The system supports ballistic motion of falling objects and models grasped bodies with a spring-damper scheme. Moreover, vibratory output is generated as feedback to the user.

The rest of the paper is organized as follows. Section 2 reviews the state of the art concerning physical simulation of virtual grasping. Section 3 describes the overall architecture of the system, while section 4 reports experiments performed to assess the capabilities of the system. The paper closes in section 5 summarizing the work.

2. Related work

A number of authors have proposed interactive virtual environments where grasping is achieved through constraint-based modelling. Huagen et al. [HLGP04] [HSQ04] presented a realistic ambient for virtual grasping based on a superquadric surfaces, real human hand textures and a three layer kinematics model, which takes into account skeleton, muscles and the skin effects. The motion of the rigid skeleton layer was driven by the data captured in real-time via a CyberGlove. Sun and Hujun [SH02] proposed a framework for two-handed virtual assembly combining task-oriented virtual assembly, constraint analysis, feature/constraint manipulation during assembly and collision-free assembly planning. Wang et al. [WJJI04] investigated an enhanced constrained motion methodology to simulate multiple constraints used in assembly operations. Yanbin et al. [YHJD07] designed a virtual training system exploiting a 5DT Data-glove, with emphasis on delivery simulation.

More advanced approaches make use of physically–based simulation for virtual grasping. The early work of Bergamasco et al. [BDB94] described a point based approach for virtual grasping. A force vector is applied to the grasped objects which includes components of static friction, dynamic friction, normal contact forces and external forces. Calculations were based on grids of control points placed on palmar sides of the phalanges and palm. Kwok et al. [KSB98] presented a hybrid control approach combining both kinematics and dynamics methods at different stages of picking to simulate accurate hand interactions in real time. The system has been experimented on simple geometric objects with a CyberGlove. Jayaram et al. [JJW*99] developed the VADE system for assembly design. VADE supports both constrained motion simulation and dynamic simulation for assembly assistance. Zaeh et al. [ZEPS04] proposed a new approach for physically–based grasping combining the GJK algorithm for collision detection with linear complementarity techniques to model and simulate multi-rigid-body dynamics with contact and friction between the user's hand and the manipulated objects. The aim of the proposed approach was to overcome the lack of force feedback of general purpose digital gloves. Howard and Vance [HV07] presented an assessment of haptic desktop system for evaluating assembly operations. A PHANTOM haptic device was adopted for human interaction. Hirota and Hirose [HH03] adopted a manipulation system using point-based collision response forces from a large number of points on the hand model. Borst and Indugula [BI05] addressed the problem of visual interpenetration of hand and object models, and performed force rendering for force feedback gloves in a single framework. The proposed approach couples tracked hand configuration to a simulation-controlled hand using a system of linear and torsional spring-dampers.

The problem of synthesis and simulation of realistic grasps has been considered in many studies in the robotics community. We briefly review some contributions where robot grasps are not generated in real-time from user interaction but are automatically planned. Sanso and Thalmann [ST94] developed a hand control system for automatic grasping. Steffen et al. [SHR07] introduced a method that combines the advantages of geometry-based and contact-based grasping. The algorithm dynamically includes previously acquired knowledge to adapt the grasping motion to the actual task. The grasping experience is formed by a database of hand postures which previously led to successful grasps. Boulic et al. [BRT96] proposed a hybrid approach for grasp synthesis where the configuration of the virtual hand is derived from virtual collision sensors.

3. System overview

The general architecture of the system described in this paper is shown in figure 1. The user interacts with the virtual environment wearing a dataglove and motion tracker. Motion of the human hand is mapped to a simulated virtual hand which can grasp objects. Visual feedback is provided by displaying the simulated environment on a desktop screen. The virtual environment is built upon a graphical engine and a physics engine. The adopted dataglove is a CyberTouch (by Immersion Corporation, Inc.), while the tracking device is a FasTrak (by Polhemus, Inc.). The CyberTouch is a tactile feedback instrumented glove with 18 resistive sensors for bend and abduction measurements. The glove comprises two bend sensors on each finger, four abduction sensors, plus sensors measuring thumb crossover, palm arch, wrist flexion and wrist abduction. Flexion angles of the distal joints of each of the four fingers are estimated via software interpolation. The CyberTouch is also equipped with actuators for vibrotactile feedback. One vibrotactile actuator is located on the back of each finger, a sixth actuator is located in the palm. Each actuator can be individually programmed to vary the strength of vibrations. Actuators are selectively activated whenever the collision detection algorithm detects contact between a region of the virtual hand (finger or palm) and one of the graspable objects in the scene. The amplitude of the vibration is set proportional to the penetration depth. The

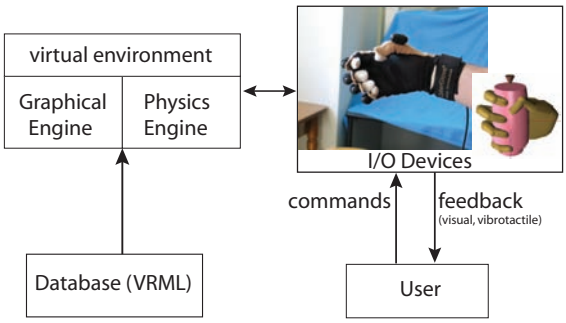

Figure 1: *System Architecture.*

FasTrak is an electro-magnetic sensor that tracks the position and orientation of a small receiver, mounted on the wrist of the CyberTouch, with respect to a fixed transmitter. The device provides accurate measurements up to $1.5m$ of distance from the transmitter.

The virtual environment includes a fully articulated model of the human hand with 22 degrees of freedom. The graphical engine has been developed upon OpenGL and the Cosmo3D API for handling the scene graph. A database of VRML is used to build at run time the virtual environment which also handles hierarchical structures defined in the VRML specification. The simulated environment allows the user to change the point of view of the scene by moving the mouse. The current setup does not support a head mounted display.

The structure of the developed applications is reported in Algorithm 1. Each rigid body is represented by constant properties such as geometry, mass, center of mass and inertia matrix. The current position of each body is encoded in a transformation matrix. Linear and angular velocities describe the dynamic state of the object. After an initialization phase where the graphical and the physical engines are initialized the simulation is started. In each cycle the system computes the actual forces and moments acting on the bodies, then the virtual environment is rendered according to its current configuration. New data are collected from the sensing devices and vibratory feedback is generated after collision detection. Temporary contact joints are created to simulate interaction between colliding bodies. The physics simulation has been developed upon the Open Dynamics Engine (ODE, http://www.ode.org/). ODE is a stable open source engine which provides realistic simulation of rigid objects. ODE uses a first order semi-implicit integrator. Constraint forces (applied to bodies to keep the constraints together) are implicit, while the "external" forces (e.g. forces applied by the user) are explicit. Moreover, the physics engine supports the definition of different types of joints.

The behavior of colliding bodies is simulated by creating hard contact constraints whenever objects collide

Algorithm 1 Algorithmic structure of the physics-based application

```
Initialize I/O devices
Import objects form database
Initialize dynamics environment
Create dynamic bodies associated to graphical objects
Set initial state (position, velocities) of all bodies
Create any joints connecting bodies
Initialize collision detection
while simulation_is_active do
    Apply forces and moments to the bodies as necessary
    Render the virtual environment
    Read data from glove and tracker
    Compute hand kinematics and velocity
    Invoke collision detection
    Analyze grasping state
    if object_is_grasped then
        Constrain object to virtual hand
    else if object_is_released then
        Set initial velocity of the object
    end if
    Send vibratory feedback to the glove
    Create a temporary contact joint for every collision point
    Take a forward simulation step
    Remove all temporary contact joints
    Advance the time of simulation time = time + △t
end while
```

with one another. The adopted collision detection algorithm is OPCODE (Optimized Collision Detection, http://www.codercorner.com/Opcode.htm). OPCODE detects collisions for models represented by either geometric primitives or arbitrary triangle meshes. The algorithm uses an optimized hierarchy of axis aligned bounding boxes to detect accurate collisions. ODE also supports resting contact with friction. The most important parameter of the simulation is the time step for the integration. In the current setup the time step has been set to $5ms$. ODE supports additional parameters that model the material properties of simulated objects, e.g. the coefficients of friction and the elasticity of collisions. Moreover, ODE has two specific simulation parameters that must be set: the ERP (error reduction parameter) and CFM (constraint force mixing). ERP indicates what proportion of errors regarding joints and contacts has to be fixed during the next simulation step. The CFM parameter specifies the admissible violation of the original constraint equation of the rigid-body system. Parameters must be regulated in order to achieve a realistic and stable simulation.

3.1. Virtual grasping algorithm

A grasped object is in equilibrium if the resultants of all forces and moments are both zero. An equilibrium grasp may be stable or unstable. We assume that a multifingered

grasp consists of n contact points with friction. Coulomb's model states that $f_x^2 + f_y^2 \leq \mu^2 f_\perp^2$, where $f_\perp$ is the normal force component while f_x and f_y are the force components lying on the tangent plane and μ is the coefficient of friction. Contact forces are constrained to lie within a cone aligned with the contact normal whose span is given by $\tan^{-1}\mu$. Each contact generates a force (f) and a moment (τ) on the object which are combined in a single six-dimensional vector called wrench w:

$$w = \begin{bmatrix} f \\ \tau \end{bmatrix} \tag{1}$$

A grasp is said force closed if given any external wrench applied to the object there exists a set of contact forces that can counterbalance the external wrench. Force closed grasps are always stable. However, not all stable grasps are force closed. In the system described in this paper we have developed a virtual grasping algorithm that constrains the object to the virtual hand if the grasp is force closed. The set of all wrenches that can be resisted by a grasp if unit contact forces are applied at the contact points is called the Grasp Wrench Space (GWS). Friction cones have unit height and are usually approximated as friction pyramids of k sides in order to compute the GWS efficiently. Therefore each contact force (f) can be expressed as the convex sum of forces (f_j) on the boundary of the friction pyramid:

$$f = \sum_{j=1}^{k} \alpha_j f_j \tag{2}$$

with $\alpha_j > 0$ and $\sum_{j=1}^{k} \alpha_j = 1$. Each of the k boundary force vectors at contact i exerts a wrench on the object, which is given by:

$$w_{i,j} = \begin{bmatrix} f_{i,j} \\ \lambda(d_i \times f_{i,j}) \end{bmatrix} \tag{3}$$

where d_i is the vector from the center of mass of the object to the contact point and $\lambda = \frac{1}{r}$ is a torque multiplier. The torque multiplier is set equal to the inverse of the maximum radius r from the center of mass. The torque multiplier makes the maximum possible torque to be unitary and therefore equivalent to the maximum assumed contact force. The use of the torque multiplier has been assumed in accordance with its frequent use in literature [FC92]. Usually the GWS is conservatively approximated by bounding the sum magnitude of the contact normals to 1, namely

$$\sum_{i=1}^{n} \| f_{\perp i} \| \leq 1 \tag{4}$$

Hence the grasp wrench space GWS is given by the convex hull of the elementary wrenches:

$$GWS = ConvexHull\left(\cup_{i=1}^{n}\{w_{i,1}, \ldots, w_{i,k}\}\right) \tag{5}$$

If the convex hull contains the center of mass, i.e. the origin of the wrench space, then the grasp is force closed.

4. Assessment of physics-based interaction

This section reports a preliminary analysis of the capabilities of the virtual environment. The system supports grasping of complex objects represented as triangle meshes as well as primitive geometric objects such as boxes and spheres. An example of a virtual grasp of a triangle mesh is shown in figure 2 where the user is grasping the Stanford Bunny.

4.1. Ballistic motion

The virtual environment enables ballistic motion of objects. Ballistic motion is applied whenever an object falls under the effect of gravity. To achieve a realistic behavior of thrown objects the system computes the translational and rotational velocity of the hand and once an object is released the initial velocity of the object is set equal to the velocity of the hand at that instant. Let $r = (x(t), y(t), z(t))$ be the position of a reference frame A local to the virtual hand. The translational velocity of the hand is given by $\dot{r} = (\dot{x}(t), \dot{y}(t), \dot{z}(t))$. The rotational velocity is computed, as required by ODE, in fixed angle representation. The vector of rotational velocities $\omega_N = (\omega_x, \omega_y, \omega_z)$, expressing the rotational velocity of frame A with respect to the fixed reference frame O, can be computed as $\omega_N = N(\Phi)\dot{\Phi}$, where $\Phi = (\gamma, \beta, \alpha)$ is the vector of RPY angles of frame A with respect to O. By recalling that ${}^O_A\dot{R}\,{}^O_A R^T = S({}^A\omega_N)$, where ${}^O_A R$ is the rotational matrix relating frame A to frame O and S is a skew symmetric matrix, it is possible to show that:

$$\omega_N = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} cos(\alpha)cos(\beta) & -sin(\alpha) & 0 \\ cos(\beta)sin(\alpha) & cos(\alpha) & 0 \\ -sin(\beta) & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\gamma} \\ \dot{\beta} \\ \dot{\alpha} \end{bmatrix} \tag{6}$$

Figure 3 shows an example of a simulated task using ballistic motion. Here the user grasps a sphere and then throws it against a wall of boxes for several times. After the impact some of the boxes collapse on the ground. When the sphere hits the ground it starts bouncing due to the elastic properties of the contacts between the objects and the ground.

4.2. Spring-damper model

Since grasped objects are constrained to the virtual hand a six degrees of freedom spring-damper model has been developed to guarantee an approximated dynamic simulation of bodies being handled by the virtual hand. This impedance approach generates forces and torques from sensed motion data. In particular, the system computes the virtual displacement, in terms of position and velocity, between the virtual object being grasped and a dynamic model of the same object which is simulated by the physics engine. The force and torque equations used to drive the dynamic model are:

$$\begin{aligned} F &= k_T d - h_T v \\ \tau &= k_R \theta - h_R \omega \end{aligned} \tag{7}$$

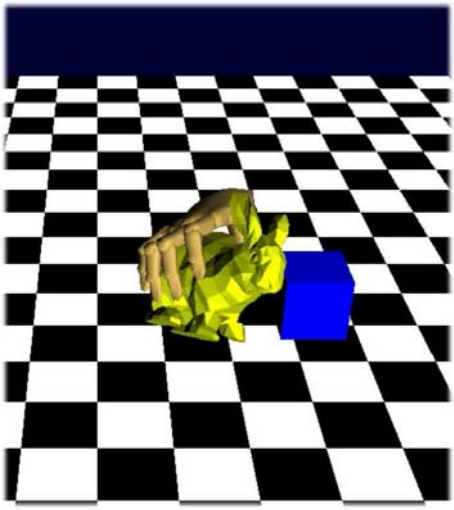

Figure 2: *Grasping an object modeled as a triangle mesh.*

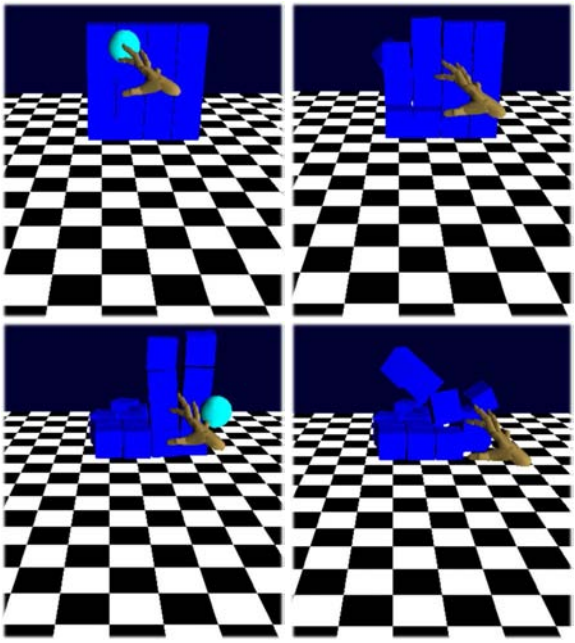

Figure 3: *Example of task exploiting ballistic motion.*

where (k_T, h_T) are the translational stiffness and viscosity constants while (k_R, h_R) are the rotational stiffness and viscosity constants. Vectors (v, ω) are the relative translational and angular velocities of the dynamic object with respect to the constrained object, while d is the translational displacement and θ is the angular displacement expressed in equivalent axis-angle notation. Figure 4 shows an example of the spring-damper effect. In the proposed experiment the user grasps a green box which acts as a tray for supporting another object (red box). The supporting object is dynamically simulated by applying the spring-damper model. The behavior of the supported object reacts to the motion of the underlying tray.

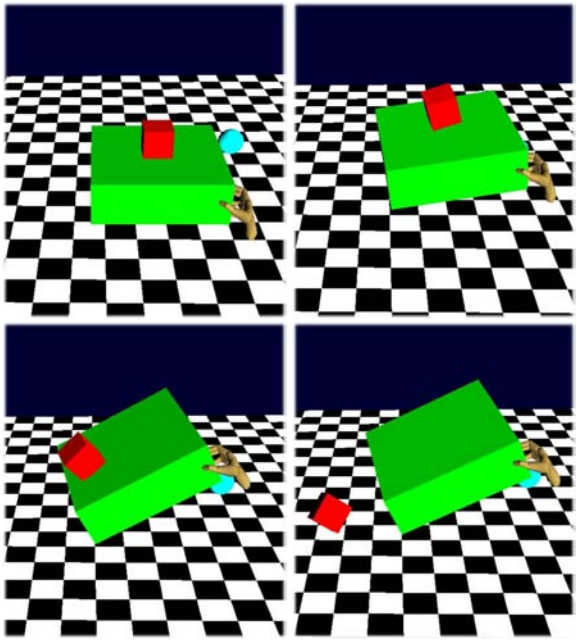

Figure 4: *Spring-damper approach for dynamic object coupling.*

4.3. Virtual lift

This section reports an example of physical simulation with bodies connected by joints. Figure 5 shows the environment which consists of a virtual lift (red box) and a blue cube which can be grasped. Initially the platform is at rest with an external force that counterbalances its gravitational force. Once the cube is released over the lift the platform starts sliding downward. The sliding motion is caused by the weight of the cube. The vertical constrained motion of the lift is obtained by connecting the platform to the ground with a slider joint. If the cube is grasped again and removed from the platform, the lift returns back to its original location.

5. Conclusions

In this paper a desktop virtual reality system has been presented. The system enables real-time user interaction and physically-based simulation of rigid objects. The graphical output of the virtual environment is managed by a scene graph engine, while physics animation is based on the Open Dynamics Engine. Advanced devices have been integrated

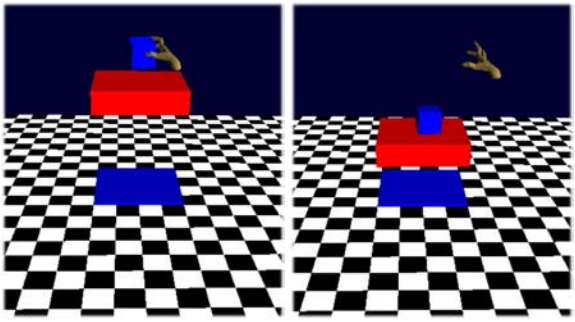

Figure 5: *Virtual platform activated by object weight.*

such as dataglove and a motion tracker which allow full hand pose estimation and natural user interaction. A 3D model of the human hand is driven in real-time in the virtual environment. Collision detection and vibratory feedback have been successfully integrated. A virtual grasping algorithm ensures stability and allows objects to be grasped. The system handles objects described as triangle meshes and supports ballistic motion of falling objects. The physical behavior of grasped objects is simulated by an impedance controller. Future work will target full dextrous manipulation of grasped objects.

Acknowledgment

This research is partially supported by Laboratory AERTECH of Regione Emilia-Romagna, Italy.

References

[BDB94] BERGAMASCO M., DEGL'INNOCENTI P., BUCCIARELLI D.: A realistic approach for grasping and moving virtual objects. In *IEEE/RSJ/GI International Conference on Advanced Robotic Systems and the Real World, IROS* (1994), pp. 3489–3494.

[BI05] BORST C. W., INDUGULA A. P.: Realistic virtual grasping. In *IEEE Conference on Virtual Reality* (2005), pp. 2938–2943.

[BRT96] BOULIC R., REZZONICO S., THALMANN D.: Multifinger manipulation of virtual objects. In *ACM Symposium on Virtual Reality Software and Technology* (1996), pp. 67–74.

[FC92] FERRARI C., CANNY J.: Planning Optimal Grasps. In *IEEE Intl Conference on Robotics and Automation, (ICRA)* (Nice, France, May 1992), pp. 2290–2295.

[HH03] HIROTA K., HIROSE M.: Dexterous object manipulation based on collision response. In *IEEE Conference on Virtual Reality* (2003).

[HLGP04] HUAGEN W., LUO Y., GAO S., PENG Q.: Realistic virtual hand modeling with applications for virtual grasping. In *ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications in Industry* (2004), pp. 81 – 87.

[HSQ04] HUAGEN W., SHUMING G., QUNSHENG P.: Virtual grasping for virtual assembly tasks. In *IEEE International Conference on Image and Graphics* (2004), pp. 448–451.

[HV07] HOWARD B. M., VANCE J. M.: Desktop haptic virtual assembly using physically based modelling. *Virtual Reality 11*, 4 (2007), 207–215.

[JJW*99] JAYARAM S., JAYARAM U., WANG Y., TIRUMALI H., LYONS K., HART P.: Vade: a virtual assembly design environment. *IEEE Computer Graphics and Applications 19*, 6 (1999), 44–50.

[KSB98] KWOK L., SUN H., BACIU G.: Physics-based virtual-hand picking in robotic manipulation. In *IEEE International Conference on Systems, Man, and Cybernetics* (1998), pp. 3489–3494.

[SH02] SUN H., HUJUN B.: Two-handed assembly with immersive task planning in virtual reality. *Virtual Reality 6*, 1 (2002), 11–20.

[SHR07] STEFFEN J., HASCHKE R., RITTER H.: Experience-based and tactile-driven dynamic grasp control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems* (2007), pp. 2938–2943.

[ST94] SANSO R. M., THALMANN D.: A hand control and automatic grasping system for synthetic actors. In *Computer Graphics Forum* (1994), pp. 167–177.

[WJJI04] WANG Y., JAYARAM U., JAYARAM S., IMTIYAZ S.: Methods and algorithms for constraint-based virtual assembly. *Virtual Reality 6*, 4 (2004), 229–243.

[YHJD07] YANBIN P., HANWU H., JINFANG L., DALI Z.: Data-glove based interactive training system for virtual delivery operation. In *IEEE Workshop on Digital Media and its Application in Museum and Heritage* (2007), pp. 383–388.

[ZEPS04] ZAEH M. F., EGERMEIER H., PETZOLD B., SCHMID H.: Haptic interaction with a glove interface in a physics based virtual environment. In *International Conference on Artificial Reality and Telexistence* (2004).