# Dynamics in Interaction on the Responsive Workbench

Michal Koutek and Frits H. Post

Delft University of Technology
*email:* {*m.koutek, f.h.post*}*@cs.tudelft.nl*

**Abstract.** In this paper we present a different view of user interaction with virtual worlds. We start from the question: how can we bring more natural object behavior into virtual environments? Currently, objects in VR applications often behave in a very un-natural way. Incorporation of physical laws in the virtual environment, together with monitoring natural user actions and behavior is desirable. We present some principles of physically more realistic behavior of virtual objects and a set of user input techniques suitable for semi-immersive VR devices such as the Responsive Workbench. We introduce springs as a new tool for assisting direct manipulation of objects in VEs.

## 1 Introduction

In user interaction with virtual worlds, consistent and realistic behavior of objects is very important. We want objects to respond in a natural and predictable way to our actions. But usually VE objects are weightless and unsubstantial, and they move without friction or inertia; this leads to altogether 'unphysical' behavior and unpredictable responses, especially in semi-immersive environments such as the Responsive Workbench (RWB), where real and virtual worlds co-exist, and should follow the same natural laws.

Absence of weight and substance may sometimes be desirable while inspecting an object or flying through an environment. But especially in manipulation tasks, mechanically realistic behavior can help to achieve consistency and predictability.

A way to provide mechanical responses from a virtual world is through haptic devices [8–11]. But haptic devices usually give force feedback through an intermediary device with a limited range, and technology for free-field haptic interaction, where objects may be touched anywhere in space, still does not exist. Therefore, we want to avoid the use of a force-based haptic interface.

We will look for a limited set of physical properties and laws of behavior to make user interaction more predictable and intuitive. We will introduce the concepts of force, inertia, gravity, contact, surface friction, and damping into the virtual world. Thus, we will have to provide a visual interface to replace direct force input. We will do this by the use of springs attached to objects, based on the following assumptions:

- a linear relation of force with spring compression / extension is intuitively understood and visualized by the spiraling shape of a spring. Thus, even without exerting real force, a user has an intuitive notion of transforming a change of spring length to a force.

- the relation between object mass and size is also intuitive, but application dependent. Specific mass of objects should be user specified. A massless world can always be created by setting all masses to zero.
- the table top of the RWB provides a natural ground plane for objects at rest.
- stability is introduced by friction and damping, reducing excessive effects of input actions on objects, and reducing undesired oscillations.
- physical contact of objects is intuitively equivalent with geometric intersection, and sound can be used to provide contact feedback.

We will propose a set of dynamic interaction primitives with objects, based on object selection and actions such as *lifting/dropping, pushing/pulling,* and *throwing/catching.* We hypothesize that dynamics in interaction will provide intuitively consistent and predictable behavior of the objects in the virtual world, and that interactive manipulation will therefore be easier and more natural. We will demonstrate the utility of the concepts from an initial implementation of a micro-world on the RWB table top bounded by walls around, consisting of spheres and boxes which can be manipulated using springs as manipulators and dynamics to govern behavior.

In this paper, we first present related work and provide an overview of 3D interaction techniques in VR. Next, we describe some possible interaction techniques for the Responsive Workbench. We concentrate on the use of dynamics in user's interactions. On the current implementation of a micro-world we demonstrate the suggested dynamic interaction in VEs. Finally, we present areas for future work.

## 2 3D Interaction Techniques

Interaction plays a very important role in virtual environments. Much has been published about interaction techniques in VR but the quest for truly intuitive and natural interaction techniques is still going on.

### 2.1 Related Work

We have studied interaction techniques used in fully immersive (CAVE, HMD) and semi-immersive virtual environments (RWB). Surveys of interaction techniques for VR workbench can be found in [1–3, 5]. New methods for remote translations in immersive VE, especially the CAVE, are presented in [4], but some of these methods can be used on the Workbench as well. In this paper we will not discuss navigation problems and we will concentrate on selection and direct manipulation of objects.

The problems of mechanics and dynamics have been studied extensively from many different points of view. It is outside the scope of this paper to review this field. We do not have the intention to implement a fully realistic dynamic manipulation system, but rather a limited set of basic physics to assist the manipulation of objects in VE.

In [6], a virtual world is described for learning Newtonian mechanics. Basic physical laws have been implemented into a virtual world, addressing the potential value of virtual reality for science education. However dynamic factors are not applied to user interaction. Another implementation of virtual mechanics is AERO [7], which is a simulation and animation system of rigid bodies. Springs are used in this system for collision processing and for elastic object connection. Elasticity and damping factors are also discussed in this paper.

In the VR literature, there is not much published work on incorporating dynamics in user interaction. Usually, haptics is used for this purpose, [8–11]. We introduce springs as a new tool for direct manipulation of objects in VEs. Many people have done lab-exercises in physics with practical experiments using springs (eg. lifting of an object attached to a spring). It has been done in the real world with dynamic feedback. In this paper we will use this as a metaphor to improve the user's feeling of mechanics and dynamics in virtual environments.

### 2.2 Interaction Techniques - Overview

We begin with a brief overview of selection and manipulation techniques in virtual environments. After that we will suggest some innovations. As we will focus on the RWB environment, navigation is of little importance. First we will discuss selection techniques, [1], [2].

*Direct picking* is the most intuitive and easy way of object selection when user has to reach an object with the pointer. When the object is not within the reach of users hand or pointer another technique must be used. In *ray casting,* a ray shoots out from user's fingers or pointer and intersections with objects are evaluated. *Gaze-directed* selection, user can select an object by looking at it. The *pointing* technique allows the user to select object by pointing at it with finger or pointer and an invisible ray shoots out from between user's eyes and pointer. *Virtual hands* give user the possibility to reach distant objects by extending virtual hands faster then the user's real hands.

Once an object has been selected, the user can manipulate it [1, 4]. When an object is out of reach, *close manipulation* brings this object near to the user. *Popping* brings a distant object into user's hands and after manipulations the object goes back to its position. *Copying* brings a copy of the object into user's hands, while the original distant object follows the manipulations performed with the copy. *Distant manipulation* allows user to manipulate distant objects with tools at a distance. In *tele-manipulation,* the user manipulates distant objects just as they were close to his body. This technique can be used in combination with virtual hands.

With the *slave method,* the manipulated object follows translations of the pointer. A disadvantage is that for larger translations, the user must perform a series of translate-release actions. The *stick method* connects user's hand or pointer with object by a ray - stick. The object is attached to this stick and follows translations and rotations (center of rotation is the user's hand). With the *3D cross-hair* method, the user can translate an object by dragging the ray of the pointer along one of the cross-hair axes.

For controlling the velocity of objects *fly* or *throttle* methods can be used. With the *fly method,* the direction and the velocity of a pointer is applied to the object. The *throttle method* uses a metaphor of a motorcycle throttle grip. By rotating the pointer about the direction of motion (the direction where points the pointer) the velocity of motion is indicated. Forward and backward motions are derived from the direction of rotation.

For use with dynamic interaction, selection and manipulation techniques may be adapted. We will now discuss the principles of some techniques which are suitable for dynamic interaction at the RWB.

## 3 Suggested Interaction Techniques

Before we describe the suggested techniques, we have to mention that our tracking system *(tracker daemon)* is able to monitor and also evaluate movements of the user. It can detect when the user is at rest, the start and the end of user's motion. In addition, some basic gestures can be recognized. On the top of this, more complex interaction techniques can be built. A specification of our RWB environment and the tracking system is given in section 5.1.

We will first describe our suggested set of selection techniques for dynamic interaction. For near objects we use *direct picking.* For distant objects we use *ray casting.* But if we will think of more natural selection method we should look into the real world. How do we know (or how does the arm know) which object we want to manipulate? The answer is that it can be derived from the path of the hand, or from the direction of hand motion. This information can be obtained from the tracker daemon buffer. When we want to grab an object we simply move our hand towards to it. This is the principle of the *'I want this one'* IWTO - method. The path information and current direction of the movement of user's hand are combined and in the predicted direction an invisible ray is projected and intersections with objects are evaluated. A possible selected candidate changes its color and if user really wants this object, just clicks the button on the stylus pointer. This is an intuitive technique taken from the real world, which can be extended for selecting distant objects.

After object selection we will now discuss object manipulation. In simple cases when object is within reach of the user's hand the *slave method* is used. Translation and rotation are applied directly to the object.

When the object is distant, the *stick method* together with the *ray-casting method* could be used. Together with ray-casting works also the *cross-plane method* where the object is being translated by the ray intersections in a plane with $z = const$. Similar is the *cross-surface method* where the ray intersects surfaces of other objects (eg. a landscape).

When the user wants to manipulate a distant object in his working space, he can grab the object near to him by a *recognizable gesture:* a fully extended hand bends in the elbow and moves into his working sphere. This method works like a magnet on objects. After manipulation an *object can be released* with a given speed or sent back to its original position. In this way an assembling task can be performed, when user brings objects into his working space and puts them together and finally places the assembled object on its position.

# 4 Dynamics on the Workbench

As discussed before, we wish to introduce the concepts of force, inertia, gravity, contact and surface friction to provide a virtual world in which objects behave more naturally. Objects behave according to physical laws and also the interaction is physics-based. We are using relevant laws of mechanics and dynamics. Basically, the three Newton's laws.

For object collisions, there are principles of conservation of momentum and of energy, and reversible conversion between kinetic and potential energy.

## 4.1 Why springs?

In VR it is just as easy to select and manipulate large and heavy objects as small ones. This may sometimes be advantage, but it is not according to our expectations. Therefore, we will try to give the user a feeling of weight or mass of objects.

We propose the use of springs as a link between user's hand and a manipulated object. By this, we can obtain a natural visual feedback during manipulation. When the user lifts a heavy object, the spring will extend proportionally to the object's weight. Also, acceleration and deceleration of the motion will affect the visible length of the spring.

## 4.2 Method Description

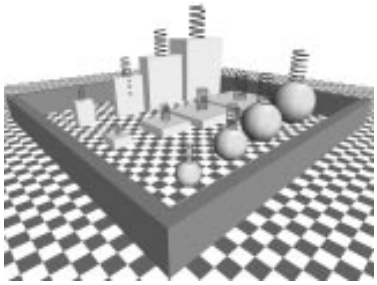
A specific mass is assigned to each object. From the volume of an object its mass is calculated. To make objects move, forces are needed. In our case, are acting *the gravity force* and *the force exerted by the user*. Counter-acting forces are *the friction force* with the surface and *the resistance force* of the air. For stability *a damping force* is implemented. We use a simplified implementation of the physics laws in our virtual world. We will ignore air-resistance. We will also only use static friction $\mu$, when user drags an object over a surface. At small speeds $\mu$ is constant. For simplicity, we will only consider linear motion here - we will not consider rotational motion, torques, and angular momentum. Here is the overview of the laws of mechanics discussed:

$$\begin{aligned}
\text{Force law: } \boldsymbol{f_a} &= m\boldsymbol{a} = m\frac{d\boldsymbol{v}}{dt} = m\frac{d^2\boldsymbol{x}}{dt^2} \\
\text{Gravity law: } \boldsymbol{f_g} &= m\boldsymbol{g} \\
\text{Friction force: } \boldsymbol{f_r} &= \mu\boldsymbol{N} = \mu.m\boldsymbol{g} \\
\text{Spring force: } \boldsymbol{f_s} &= -k\boldsymbol{x} \\
\text{Damping force: } \boldsymbol{f_d} &= -c.\frac{d\boldsymbol{x}}{dt} \\
\text{Momentum: } \boldsymbol{p} &= m\boldsymbol{v} \\
\text{Linear Momentum: } m\boldsymbol{a} &= \sum_{i=1}^{n} \boldsymbol{F_i}
\end{aligned}$$

Where: $m$=mass, $\boldsymbol{f}$=force, $\boldsymbol{x}$=position, $\boldsymbol{v}$=velocity, $\boldsymbol{a}$=acceleration, $\boldsymbol{g}$= gravity acceleration, $\mu$= static friction coefficient, $k$= spring constant and $c$= damping factor.

During user manipulation, a spring is attached to the center of mass of an object. The spring constant $k$ is calculated for each object differently. The rest length of the spring $x_0$ is also calculated automatically from the mass of an object. The result is that for heavy objects a stronger and longer spring is used, and a weaker and shorter spring for light objects. If the same spring constant and same rest length would be used for every object it would have a wrong effect on user manipulation. The user would have to lift his hand very high for heavy objects with the weak spring. To avoid this, every object has its own type of spring. Such a spring is able to hold the attached object with an extension of 20 % of its rest length $x_0$. The absolute mass of objects is application dependent and directly affects the spring constant. The difference between the springs is visible, see fig.1. Springs are normally only visible during manipulation of a selected object.

$$\text{Spring constant:} \quad k = -\frac{m.g}{x_m} = \frac{m.g}{0.2x_0}$$
$$\text{where} \quad mg = -kx_m; \quad x_m = -0.2x_0$$



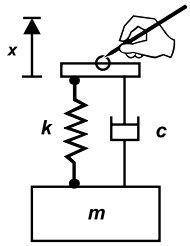



**Fig. 1.** (a) Various objects and different springs (b) Spring Damper System

### 4.3 Spring Damper

To reduce oscillations during manipulation with an object attached to a spring, we have implemented a *spring mass-damper system* [12].

$$m\ddot{\boldsymbol{x}}(t) + c\dot{\boldsymbol{x}}(t) + k\boldsymbol{x}(t) = \boldsymbol{f}(t)$$

Here $m$ is the mass of the object, $\boldsymbol{x}(t)$ is the extension/compression of the spring, $\boldsymbol{f}(t)$ is the force acting on the object only in $\boldsymbol{x}(t)$ direction, $k$ is the spring constant and $d$ is the damping factor.

When the object is lifted in the air and hanging on the spring, we can rewrite the model this way:

$$m\ddot{\boldsymbol{x}} + c\dot{\boldsymbol{x}} + k\boldsymbol{x} = m\boldsymbol{g}; \quad (\boldsymbol{x} = \boldsymbol{x}_s + \boldsymbol{x}_d)$$
$$m\ddot{\boldsymbol{x}_d} + c\dot{\boldsymbol{x}_d} + k\boldsymbol{x}_s + k\boldsymbol{x}_d = m\boldsymbol{g}; \quad (k\boldsymbol{x}_s = m\boldsymbol{g})$$

Where $\boldsymbol{x}_s$ is the static extension of the spring and $\boldsymbol{x}_d$ is the change of extension during the motion. The differential system to solve will then be:

$$m\ddot{\boldsymbol{x}}_d + c\dot{\boldsymbol{x}}_d + k\boldsymbol{x}_d = 0; \quad \boldsymbol{x}(0) = \boldsymbol{x}_0, \quad \dot{\boldsymbol{x}}(0) = \boldsymbol{v}_0$$

More about solutions can be found in [12]. The damping factor plays a very important role. Its value is derived in a similar way as the spring constant, so that every object has its spring and damper. *Damping* refers to an energy dissipation mechanism, either intentional or parasitic, such as air friction or structural damping. The *damper* is the energy dissipating element of the system. The *damping factor* measures the ability to damp the motion of the mass. In fig.2, the effect of the damping factor can be observed.

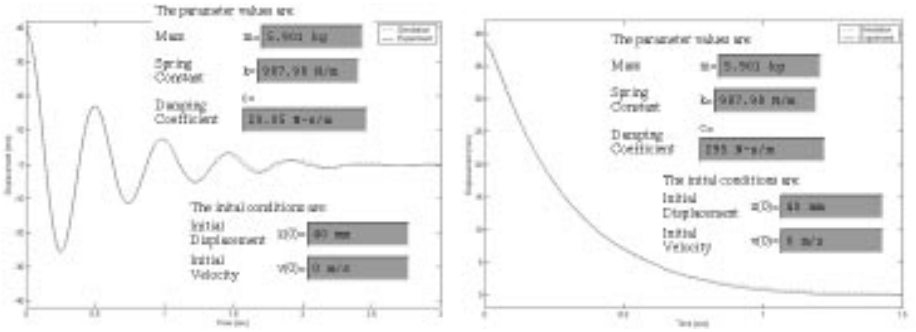



**Fig. 2.** Damping examples

### 4.4 Spring Manipulation Techniques

An object can be manipulated using a set of techniques as described below. A spiral spring is used as a handle, and the object will show its mass and inertia by its behaviour according to the laws of dynamics.

The actions are performed by selecting an object by moving the stylus pen towards it, the IWTO method, see section 3. At a close distance, a spring will be displayed between the object and the user's hand position. The spring can be extended or compressed by keeping the pen button depressed while moving the pen towards or away from the object.

Virtual forces are thus applied by the user to objects through the virtual springs, which act as displacement-to-force transducers. The user will see the extension or compression of a spring, and forces are inferred by the linear relation of displacement and force. The spring constants are calculated as described in section 4.2. The forces may be exerted on an object in an arbitrary direction. They are assumed to work on the object's center of mass, and thus the forces will induce only linear motion, and no rotation, as no torques are induced. This was done for practical reasons only. The laws governing rotational motion, torques, and angular momentum are very similar to those for linear motion, and will be incorporated later. Although friction on the ground plane can induce torques and rolling motions, these effects are ignored in the current version of the system.

The exerted forces are decomposed into lifting forces (perpendicular to the ground plane), and dragging forces (parallel to the ground plane). To ensure easy

and stable manipulation, both linear and pendulum oscillations are damped critically, which means that motion will essentially stop after one cycle. For clarity, the objects are assumed here to rest on the ground plane, although other horizontal planes will have the same effects.

Fig.3 shows the three cases: *lifting, pulling* and *pushing*. User exerts a force $f_u$ which is unknown. From the change of extension of the spring $\boldsymbol{x}_1$ we can obtain the spring force $\boldsymbol{F}_s$ which makes object attached to the spring accelerate and move.

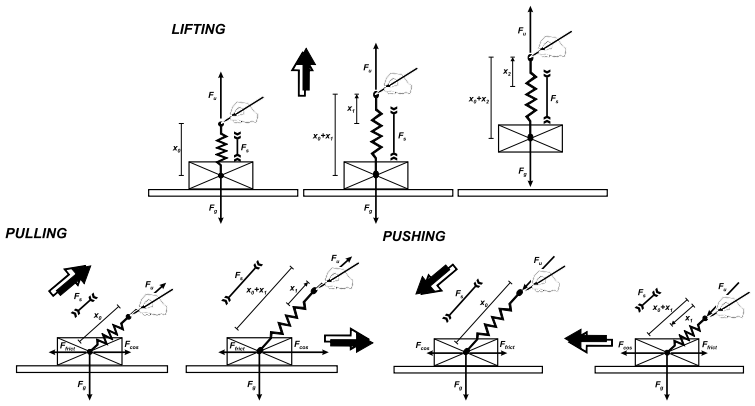



**Fig. 3.** Spring manipulation cases

The set of dynamic manipulation techniques is defined as follows:

- **lift:** pull the spring upward, until the spring force exceeds the object's weight, and the object gets an upward acceleration, counter-acted by the decrease on the force caused by the shortening of the spring length. Oscillation is damped.
- **drop:** the object is released from the spring connection and falls down until it touches the ground plane.
- **pull:** pull the spring in a horizontal direction away from the object until the force exceeds the static friction of the object on the ground surface. The object gets a horizontal acceleration and slides over the surface, counter-acted by friction force, and the decreasing force from the shortening of the spring.
- **push:** the same as pull, but the spring is compressed toward the object.
- **throw:** the object receives the initial velocity from the user's hand and is thrown in the given direction, the spring disappears. The trajectory of the object in the air is then only influenced by the gravity force and the object is falling to the ground.
- **catch:** stop the object motion by blocking it's motion path. The spring between the user's hand and the object is compressed and the object decelerates and stops moving. Oscillations are again damped.
- **swing:** a combination of lift-pull-drop. The object behaves as a pendulum on the spring; the swinging motion is damped, the object is moved through in the air, and is dropped at the chosen position.

The difference between dropping and throwing is made by analysis of hand motion data from the tracker. With dropping the pen button is released while the user's hand is at rest, for throwing the user releases the button while the hand is moving. Catching flying objects can be done by positioning the hand to block the motion path and pressing the button.

Wherever spring actions are performed (too) far from the user's body, a *'fishing rod' technique* can be used to attach and move the spring and the object.

### 4.5 Collisions and Constraints

For natural behavior in virtual environments, the collisions and constraints are essential. Of course, we cannot stop user's hand to move through objects. But we can disable user manipulation of an object through another object. We can use here again the advantages of the spring manipulation. The manipulated object will stay at the place of collision with the other object (see collision with a wall, Fig. 4) while user is still trying to pull. In this case, the pulling will only affects the length of the spring, and no motion is induced.

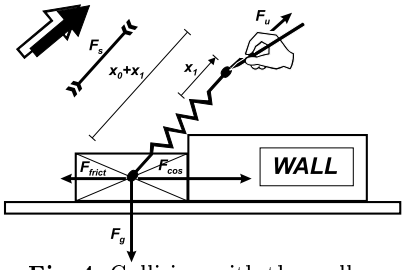



**Fig. 4.** Collision with the wall

The basic constraint is the ground plane. No object can get through it. The ground can always produce enough normal force to support any object. Calculation of collisions with the ground is the basic attribute of our mini-world. We also handle collisions between objects inside the world (spheres, boxes). Elastic collisions are simulated. We use there the law of momentum conservation and the law of energy conservation.

## 5 Experimental Application

To test the concepts of this paper we have implemented an experimental application where user can perform dynamic spring manipulation with virtual objects in a mini-world. The mini-world consists of several boxes and spheres, surrounded by walls, see Fig. 5. The basic concept of the RWB is that the user is standing in the real world, where the physical laws apply, and is partly immersed in the virtual mini-world which is projected on the Responsive Workbench. Finally, the user can also observe a visual feedback of dynamic behaviour of manipulated objects.

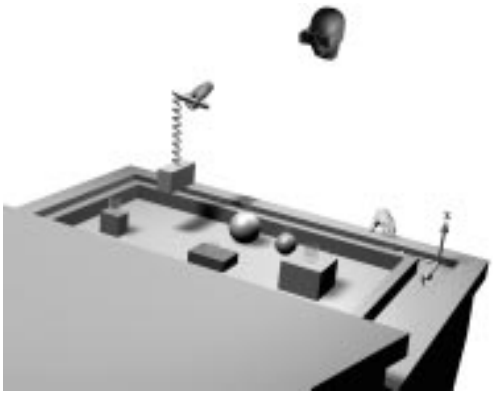


**Fig. 5.** Illustration of the dynamic manipulation with objects in the mini-world

### 5.1 RWB Environment

The VR system which we are using is based on the SGI ONYX2, with 4x MIPS R10000, 1xIR2 graphics pipe and a stereo projection table. For implementation we have used C++, Iris Performer and OpenGL. Our Responsive Workbench is equipped with a Fastrak [13] tracking system, to track the position and the orientation of user's head and hand (stylus pointer). A *Tracker daemon* reads periodically at a rate of $50Hz$ the tracking data and stores them in a buffer.

Velocity and acceleration vectors of the user's hand are calculated. During our experiments we have done measurements on the tracker data, fig.6. These data are used together with time information for calculation of the spring force and for the simulation of the spring-damper system.

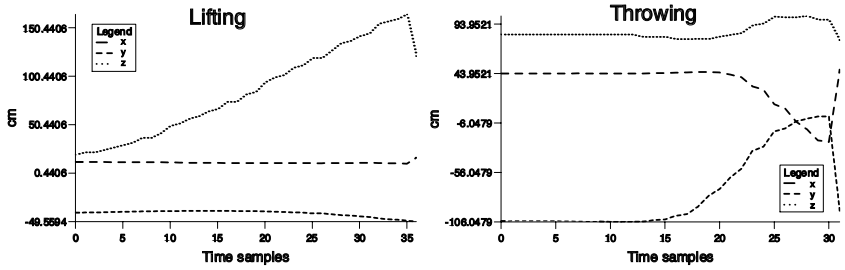



**Fig. 6.** Tracker positions during lifting and throwing manipulations

### 5.2 Dynamic Manipulation Examples

The figures 7 and 8, illustrate a procedure of dynamic manipulation of an object in the mini-world. On these pictures, a simple manipulation task is performed: 1) lifting of the object, 2) translational motion in the air and 3) dropping of the object. While the object is in the air, there is also a swinging motion.

In figures 9, 10 and 11, we show some pictures taken of the actual system implemented in the RWB environment.

**Fig. 7.** Lifting of object attached to spring

**Fig. 8.** Moving and dropping of object

**Fig. 9.** RWB-overview: the dynamic manipulation with objects in the mini-world

## 6 Conclusions and Future Work

We have developed a scheme for introduction of dynamic object behaviour in manipulation of objects in virtual environments. Initial results have shown that object's behaviour appears more natural and predictable than the 'unphysical' objects in most virtual environments. Also, it seems that the dynamics gives a coherent 'look and feel' to a 3D direct manipulation user interface. We can easily define a long list of extensions for future work in this area. The set of basic user operations will have to be extended, especially if we want to introduce rotational motions. The numerical computations for dynamic simulation can become very complex, and may put serious limits on performance. Both accuracy and efficiency of this needs more attention.

For simulation of contact behaviour, efficient collision detection and handling, and geometric constraint maintenance should be integrated. A problem which needs attention is th difference in direction between the gravity fields in the virtual and real environment, due to the angle of the RWB's table top.

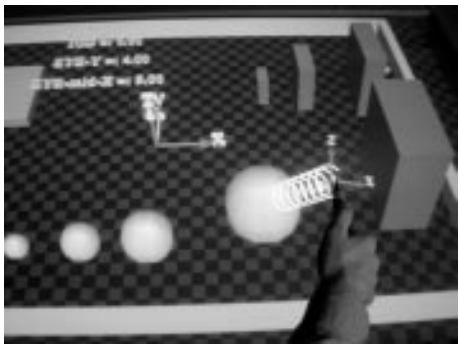


**Fig.10.** RWB-detail: Lifting of a sphere

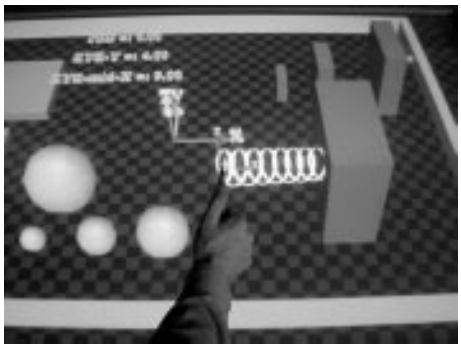


**Fig.11.** RWB-detail: Dragging of a box

## References


1. R. van de Pol, W. Ribarsky, L. Hodges, F. Post, *Interaction Techniques on the Virtual Workbench*, Proc. of Eurographics Virtual Environments '99 workshop, Springer, Vienna 1999.
2. D. Bowman, D. and L. Hodges, *An evaluation of Techniques for Grabbing and Manipulation Remote Objects in Immersive Virtual Environments*, Symposium on Interactive 3D Graphics, 1997.
3. L.D. Cutler, B. Fröhlich, P. Hanrahan, *Two-handed Direct Manipulation on the Responsive Workbench*, Symposium on Interactive 3D Graphics, 1997.
4. J. D. Mulder, *Remote Object Translation Methods for Immersive Virtual Environments*, Proc. of Eurographics Virtual Environments '98, Springer, 1998.
5. D. Bowman, L. Hodges, *User Interface Constrains for Immersive Virtual Environment Applications*, Proc. of IEEE VRAIS, 1997, pp. 35-38.
6. C.Dede, M.C.Salzman, R.B.Loftin, *The development of a virtual world for learning newtonian mechanics*, Multimedia, Hypermedia, and Virtual Reality, Berlin: Springer/Verlag, 1996, pp. 87 -106
   *NewtonWorld:* http://www.virtual.gmu.edu/newton.htm
7. H. Keller, H. Stolz, A. Ziegler, T. Braunl, *Virtual Mechanics - Simulation and Animation of Rigid Body Systems*,1996.
   *Aero:* http://www.ee.uwa.edu.au/ braunl/aero/
8. T.M. Massie, J.K. Salisbury, *The phantom haptic interface: A device for probing virtual objects*, Proc. of ASME Haptic Interfaces for Virtual Environments and Teleoprator Systems,p:295-301, 1994.
9. D.C. Ruspini, K. Kolarov, O. Hatib, *The haptic display of complex objects*, Proceedings of ACM SIGGRAPH '97, pages 345-352,1997.
10. J.M. Brown, J.E. Colgate, *Physics-based Approach to Haptic's Display* Proceedings ISMCR '94, Topical Workshop on Virtual Reality, Houston TX, 1994.
11. Grigore C. Burdea, *Force and Touch Feedback for Virtual Reality*, Publishers City, ISBN 0-471-02141-5, 1996.
12. http://links.math.rpi.edu/devmodules/mechanicalosc/springmass/
13. http://www.polhemus.com/ftrakds.htm