Eighth Eurographics Workshop on Virtual Environments (2002)
S. Müller, W. Stürzlinger (Editors)

Virtual Spring Manipulators for Particle Steering in Molecular Dynamics on the Responsive Workbench

Michal Koutek ♣, Jeroen van Hees ♠, Frits H. Post ♣, A.F. Bakker ♠

♣ - Faculty of Information Technology and Systems, ♠ - Faculty of Applied Sciences
Delft University of Technology, Delft, The Netherlands

Abstract

In this paper we present new virtual spring manipulator-based tools for steering particles in molecular dynamics simulations in virtual environments. We briefly overview the MolDRIVE system, our visualization and computational steering environment for molecular dynamics real-time simulations, which is the platform for our particle steering implementation.
Our study concentrates on visual feedback tools. We compare a basic virtual particle steering method with two other methods using a spring manipulator. The first just creates a visual feedback of a flexible connection between the user's interaction device and the steered particle, while the second technique creates a visual illusion of force feedback. The user can, through the spring manipulator, exert a force on the manipulated particle in the MD simulation. All presented particle steering tools are intuitive and easy to use.

1. Introduction

Current trends in molecular dynamics (MD) visualization show a growing importance of interactive steering capabilities for the MD simulation systems. Thanks to the increasing computational power we can simulate larger MD problems in real-time. But there will always be enough MD simulations where we would have to wait days or weeks for results. Thus many of today's MD systems [9, 10, 11] support visualization and steering of running simulations.

For many reasons, atomic or particle steering seems to be a very attractive function of MD simulation and visualization systems. It can be used for protein design [14] and for molecular docking. With the particle steering the user can make a particle overcome energy barriers within the simulated system. This way a desired configuration can be reached in a shorter time, which is very useful for studying specific energy system configurations. Based on previous research of our computational physics (CP) group [5], we perform particle steering on real-time simulations of the β-Alumina electrolyte system.

Virtual Reality systems, such as the Responsive Workbench (RWB), the ImmersaDesk or the CAVE-like systems, offer a stereoscopic 3D immersion into the microscopic scale molecular environments. The 3D tracking technology gives us the opportunity to interact with these virtual environments (VE). We can create specialized VEs for Interactively Steered Molecular Dynamics (ISMD) [13, 14], where we can control the simulation as well as interact with the particles to allow the user to get in touch with this micro-environment. Appropriate interactions and responses from the system are necessary.

Our proposed particle steering methods offer visual feedback, showing whether the performed interaction with the particle can be physically valid. Each user interaction with a particle is validated by the simulation program.

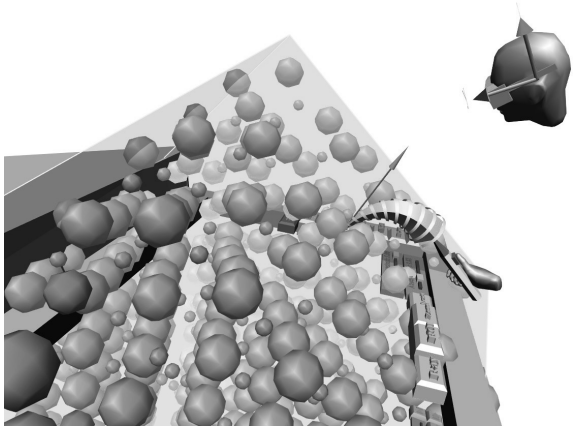

Figure 1: *Visual force feedback using spring manipulator*

Currently, haptic devices are often used to obtain an input position and force from the user, as well as to render the force feedback.

Haptics adds to VEs a dimension of touch and force feeling; the user can feel a real force during pulling or pushing objects. Haptics is also used to provide the force input and the force feedback during atomic steering [7]. Many of the PHANToM - like haptic devices are used on desktop systems and less of them are used in immersive VEs [15, 16, 17, 18]. Some of these MD visualization systems use the haptics to apply force on particles or to feel the forces of the simulated system. Advantage of VEs is that they produce greater 3D immersion and much more natural interaction with the MD simulation.

Motivation of our research is to prove our concept that visual force feedback is sufficient for effective particle steering in MD simulations. We present a graphical force-feedback method and we provide a visual interface as an emulation of direct force input. We have implemented in the MolDRIVE system the following three particle steering methods:

- **virtual particle** displays a virtual particle on a user-requested position and the original particle on a position accepted by the simulation; the new particle position is sent to the simulation and checked for validity.
- **spring feedback** displays a bending spring between the stylus and the manipulated particle on accepted position; the virtual particle with the ray shooting from the stylus show the desired position, more in section 5.
- **visual spring force feedback** displays a bending spring between the stylus and the particle; the 2 DOF (degrees of freedom) spring deformation (stretching and bending) defines an external force acting on the particle, see figure 1; the force is sent back to the simulation, more in section 6.

In this paper, we first review some related work on MD simulation in VE, on particle steering, on force feedback and on the usage of springs as a visual feedback instrument. This work follows the initial implementation of the spring manipulation tools [1, 3]. Then we provide an overview of the MolDRIVE system, and we describe in detail the three particle steering methods. This paper documents their successful implementation. Finally, we present some conclusions and areas for future work.

2. Related Work

Molecular Dynamics simulations are used to study the properties and behaviour of complex particle systems by solving Newton's equation of motion numerically for all particles for a relatively small time interval.

The growing interest for being able to steer an MD simulation has resulted in the development of several systems such as Interactive Molecular Dynamics (IMD) [7] and Steered Molecular Dynamics (SMD) [6].

We have developed the MolDRIVE system on top of the DEMMPSI simulation program. DEMMPSI [9] stands for Delft Empirical Model of Many Particle SImulations. The program is developed at Delft University of Technology and is written in C++. Because of its reasonable computational speed compared to other MD programs [5], and its parallel implementation using MPI, it is a good candidate to perform atomic steering of real-time Molecular Dynamics simulations.

Usually haptic devices [15, 16, 17, 18, 19] are used to add user force on particles and to experience the reaction of the simulation. In SMD external forces are implemented with springs between the target position and the restrained atom. Visual Molecular Dynamics (VMD) [12] is commonly used to perform the 2D or 3D visualization. VR extensions to VMD have been presented [13, 14], where the user's immersion into the VE is supported by head-tracking with ability to interact with the visualization.

The MolDRIVE uses the RWB as a Virtual Reality visualization environment with real 3D interaction. It uses a metaphor of a laboratory table, where the user performs an interactive experiment with MD system. What is specific in our case, is the use of the DEMMPSI simulation, and our new contribution is the particle steering tools, which use the spring manipulator.

The springs are very well established phenomena in the world of physics and graphics as well. Springs were used also by others for manipulation of objects on the Responsive Workbench [8]. They demonstrate the use of simple 1 DOF springs in set of multi-spring configurations to manipulate objects, giving visual force feedback. To simulate the spring behaviour they use the Coriolis physical simulation package. The simulation update of 5 Hz seems to be really difficult to use for interactive visual force feedback, which they claim to achieve.

Our spring manipulators are based on our previous work, the spring manipulation tools [1, 3], where we have demonstrated the use of spring for manipulation of objects in virtual environments in simple assembly tasks. The spring-fork [1] needs just one hand with the stylus pen to control position and rotation of attached object, and it provides a clear visual force feedback. When we look at frame rates, our simulation of spring deformation is not a bottle-neck in the visualization and interaction pipeline.

3. Molecular Dynamics Real-time Virtual Environment (MolDRIVE)

We have developed a system named MolDRIVE, which is a virtual environment for visualization and steering of real-time molecular dynamics simulations. MolDRIVE is in fact a visualization and computational steering environment with an interface to several MD simulation programs, which run in parallel on the supercomputers. Currently, it uses DEMMPSI [5, 9].

Remote simulations allow the high performance rendering machine, to fully concentrate on the visualization part of the proces and larger systems can be visualized. The system layout is shown in figure 2. The MD simulation, in our case DEMMPSI, runs remotely on parallel supercomputers such as a CRAY T3E (128 proc.), an SGI 1100/1200 Beowulf Linux cluster (54 proc.) or an SGI Origin 2000 (8 proc.).

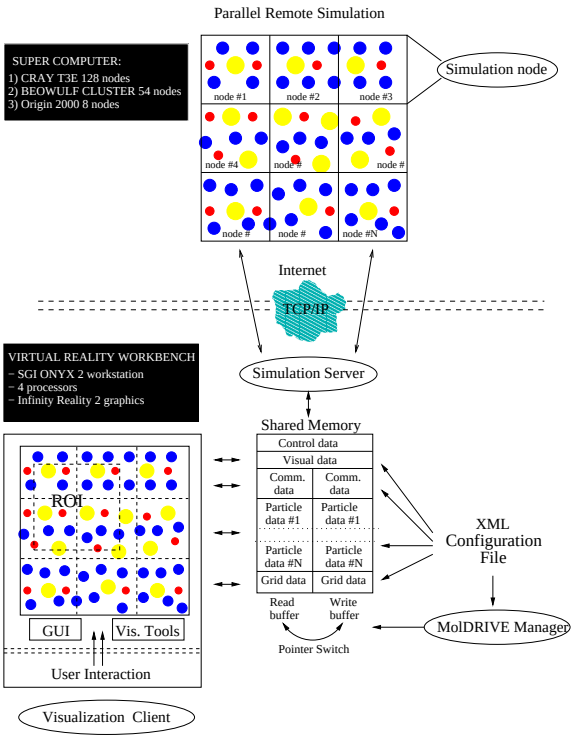

Figure 2: *MolDRIVE overview: remote parallel MD simulation running on (parallel) supercomputers; visualization and computational steering on the RWB*

3.1. System Components

Simulation server is a thread-based program, which communicates simultaneously through TCP/IP sockets with each MD simulation node on the supercomputer. The simulation server runs on an SGI ONYX 2. Each time step the server transmits a data request of the visualization client to each simulation node. Each node handles this request and sends the simulation data back to the server. This data is then placed into the shared memory. The send request is also used to transmit user interactions back to the simulation.

The Responsive Workbench (RWB) **visualization client** also runs on the SGI ONYX 2. The RWB visualization client is implemented using RWB-Library [2, 4] and is based on Iris Performer and OpenGL. It runs in parallel on all 4 processors of the ONYX 2 to maximize the speed of the visualization pipeline and to keep the interactive visualization frame rates.

The visualization client creates a graphical representation of the simulated system and updates the particle position, forces and velocity vectors. It can also visualize the derived grid data from the simulation, such as particle densities, potentials, kinetic energy, etc. Through the 3D GUI has the user access to several visualization and steering tools.

Shared memory is used to communicate and exchange simulation and communication data between the visualization client and the simulation server.

MolDRIVE manager reads an XML configuration file with the description of the simulation and the visualization environment. It initializes the shared memory data structures. It also activates and controls the RWB visualization client and the simulation clients (through the simulation server).

The shared memory consists of *particle data*, such as position, velocity and forces, *grid data* such as particle density or potential, and the *communication data*, which are used to steer the simulation and to get a feedback data. We use a double buffering to speed up the data throughput. The *control data* is used to manage the read and write access from both visualization client and simulation server to the shared memory. RWB visualization client reads the data from the *read buffer*, while the simulation server fills in the *write buffer* with simulation data for the next time step. When both have finished the MolDRIVE manager switches the pointers and tells to the visualization client that new data is ready.

Another benefit of this method is that it reduces the effects of network latency. A small disadvantage is that during particle steering the write buffer is filled with simulation data which are one step behind.

For the MolDRIVE system, we use a Na^+-β-alumina simulation as an example, which runs in real-time at reasonable frame-rates, up to 24 Hz, depending on the number of particles in the simulated system and the complexity of inner system relations and interactions. This Na^+-β-alumina electrolyte is studied because of it's fast ionic conduction properties. We usually run this simulation on 2, 4 up to 8 nodes in parallel on the supercomputer. The simulation scales up quite acceptably with the number of processors.

DEMMPSI uses domain decomposition of problem space; thus, each processor has to simulate only the particles which fall within its sub-domain. In case of long range interaction forces with particles from other sub domains communication with other simulation nodes takes place. For systems containing 2000 up to 4500 particles, we have experimentally found that 8 nodes seems to be an optimal number for DEMMPSI. If more processors are used, communication between the computational nodes slows down the whole simulation. From our experience, if the refresh rate does not drop much below 8 Hz, we can still interactively perform the particle steering even with a system containing more than 2000 particles.

On top of the MolDRIVE system we have implemented three particle steering methods, which we will describe in the next three sections.

4. Virtual Particle Steering Method

The virtual particle method and its enhanced version with the spring feedback send a newly requested particle position through the control data buffer via the simulation server to the simulation nodes. The simulation then evaluates the new position in a new simulation time step. If this new position doesn't destabilize the simulated system, the new data are sent to the simulation server and then placed into the shared memory where the visualization client reads its data.

4.1. Underlying Physics

During manipulation of a particle through the system, the potential energy of this particle changes and may come to a value which the particle normally could not get. Such a situation occurs for example when the user tries to place a particle on very small distance from another particle. This could cause unpredictable behaviour and eventually resulting in a simulation crash. To prevent this, the Boltzmann factor is used formula 1. It gives us a relative probability P, that the system will move to a configuration with an energy value at a distance ΔE from the current energy.

$$P = exp(\frac{-\Delta E}{k_B T}), \qquad \Delta E \geq 0 \tag{1}$$

k_B is the Boltzmann constant and T is the temperature of the system. For the reference energy the particle energy at the first position of the interactive movement is used. A new particle position is considered to be valid if the corresponding energy at that position could occur with a probability that is larger than or equal to a probability α, which is usually chosen to be in the order of 1%, which is a commonly used value .

4.2. Practical Usage

The virtual particle steering method is shown in figure 3. The user selects a particle using the ray-intersection. At the moment of the first click on the stylus button, the transformation from the stylus pen to the particle is computed. Next time, when the RWB-library updates the position and orientation of the stylus pen, the new position of the particle will be computed using this transformation. At this position, at the end of the ray, a virtual (blue) particle is displayed. It indicates a new desired position of the particle. This position is sent to the MD simulation for validation. The white particle shows this particle on the last valid position that was accepted by the simulation program.

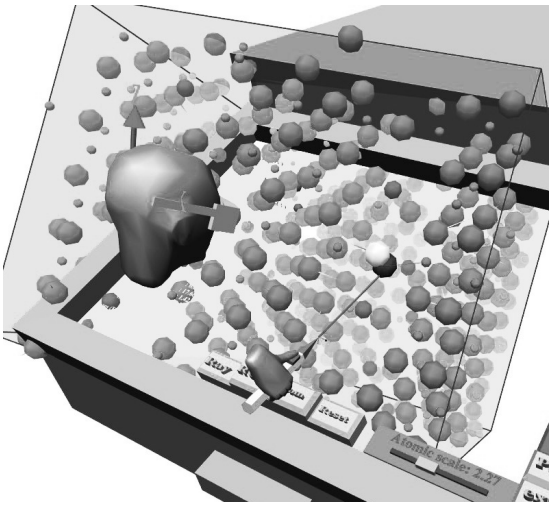

Figure 3: *Virtual particle method: overview*

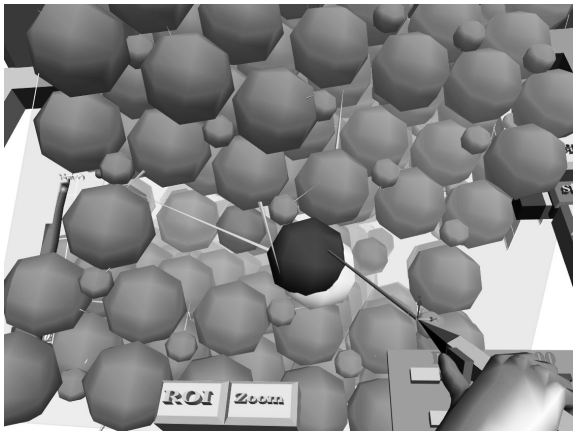

Figure 4: *Virtual particle method: detail*

Figure 4 shows the steering of the Na^+ atom through the conduction layer of the electrolyte. The velocity of particles as well as the forces acting on them are visualized using arrows. They react to the user's interaction giving an immediate visual response.

The virtual particle always follows the motions of the stylus, while the accepted white particle only moves when the simulation allows this. Sometimes it results in a very inconsistent view. For a user manipulating a particle, it seems as though the original particle was forgotten at its place and some new particle pops up and follows the ray. With increasing distance from the white particle also the visual gap between them increases.

This is the situation where the spring manipulator can be used. It will connect the stylus with the manipulated particle on its valid position.

5. Spring Feedback Particle Steering Method

This method visually differs from the previous one in the way the user experiences the acceptance of the new position by the simulation. During particle manipulation, the user's stylus pen is always connected to the selected particle through a spring manipulator, as shown in figures 5 and 8. The position of the manipulated particle is equivalent to the last position accepted by the MD simulation. The acceptance is calculated in the same way as in the previous method.

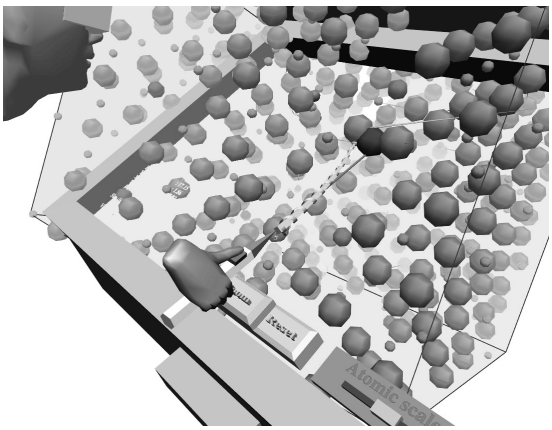

Figure 5: *Spring feedback method: overview*

This spring based interaction tool has 2 DOF for deformation: stretching and bending. Next it has 6 DOF in position *XYZ* and orientation *HPR* (Head, Pitch and Roll), which is given by the tracking of the stylus pen. The spring manipulator is attached to the tip of the stylus pen and points always in the Z-direction of the stylus's local coordinate system. The other end of the spring manipulator is always attached the manipulated particle at the point of the initial ray intersection.

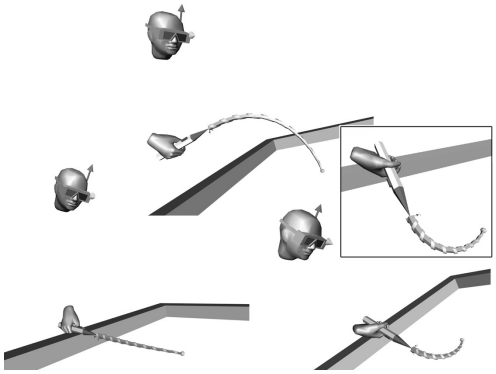

Figure 6: *Spring manipulator: experimental deformations*

At the moment of the first button click, a transformation from the center of the particle to the point of initial intersection is calculated, see Figure 10. Next time, when a new accepted particle position comes and the RWB-library reads the new stylus position and orientation, the new intersection point is recalculated by transforming the actual particle central position with the pre-computed transformation. The spring manipulator will bend and stretch to end precisely at the new intersection point, see Figure 10.

This scheme is similar to the original spring-fork [1] with 3 deformation DOF (stretching,bending,torsion). The spring-fork can control position and orientation of objects with 6 DOF, see Figure 7.

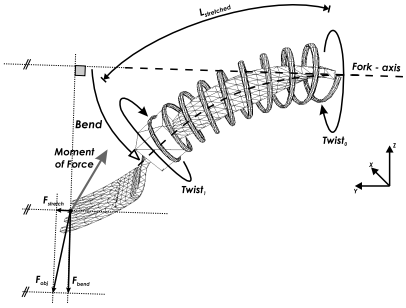

Figure 7: *Original spring-fork manipulation tool*

The new spring manipulator has only 2 deformation DOFs. It bends without torsion and controls only position of an object. This tool has only a single point attachment with the particle. Thus rotation cannot be applied to particles and for single particle steering has also no need.

The effect is the user never loses a particle during manipulation. It stays visually connected to the stylus, no matter how the user deforms the spring, see Figures 6, 8 and 9.
The new (desired) position which is sent to the simulation is indicated by a virtual particle at the end of the stylus ray just as in the previous method. The spring metaphor makes the user feel that the particle is being dragged to a new position, which is experienced as more natural than the previous steering method.

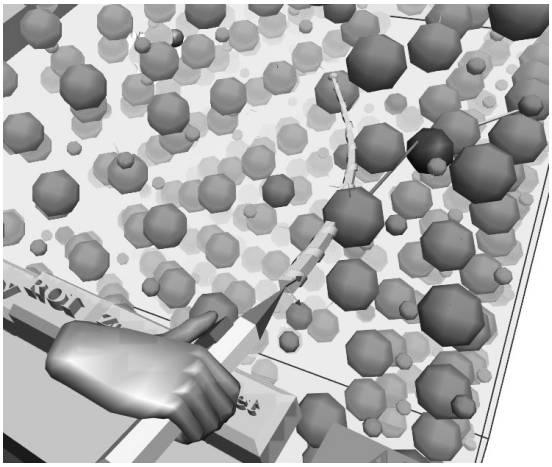

Figure 8: *Spring feedback method: detail*

6. Spring Force Feedback Particle Steering Method

The previous method provides only a visual contact with the manipulated particle. The feedback is provided based only on particle positions. Sometimes, it results in irregular motion of the particle, as the new particle positions may be accepted or rejected by the MD simulation.

A way to avoid this is to use the force input and the force feedback delivered by the spring instead. The spring manipulator provides visual force feedback as well as an emulation of direct force input through the idea of bending and stretching a spring.

6.1. Model of the Spring Manipulator

The graphics model of the spring manipulator consists of a spring, a beam column inside and a little ball at the tip of the instrument, which supports the idea of a single point contact with the manipulated particle, in contrast to the original spring-fork, which has contact with intersected objects at four points, and therefore can control also rotation of objects. This was not needed for a single particle steering. Thus the spring manipulator became more simple. As mentioned before, the spring has 2 deformation DOFs; stretching and bending. Figure 9 shows in wireframe: a) extended spring b) regular bending c) highly compressed spring d) extreme spring deformation.

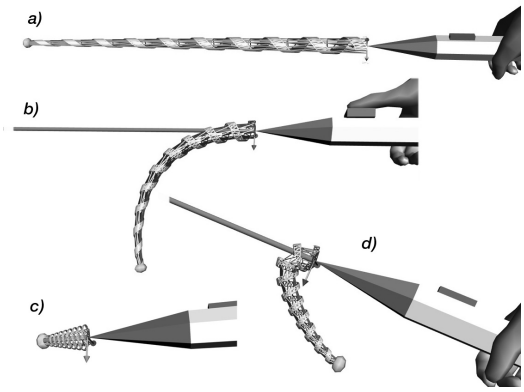

Figure 9: *Spring deformations in wireframe*

The geometric model of the spring manipulator is kept simple, with a small number of vertices, because the vertices and normals of the facets have to be transformed according to the spring deformation. This happens each time when the RWB-library reads a new position and orientation of the stylus (48Hz). Spring deformation follows the arc-length parameterized cubic Bézier segment, which is defined by the 4 control points: A,B,C,D, see Figure 10. More about deformation model of the spring tools can be found in previous paper [1].

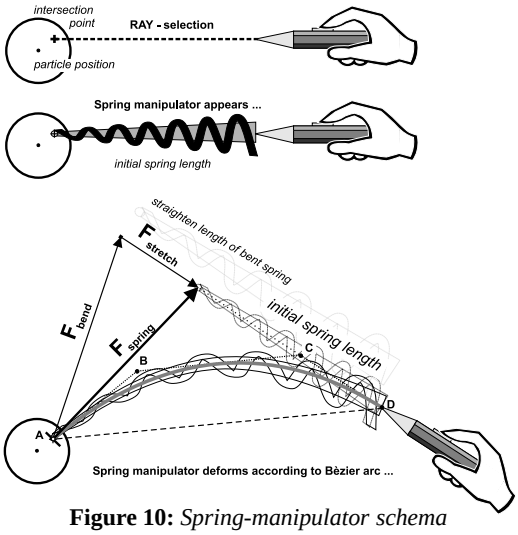

Figure 10: *Spring-manipulator schema*

Stretching and bending of the spring manipulator causes a reaction of exerting a force to restore its initial length and straight form. The total spring force consists of the stretching and the bending component:

$$\text{Spring stretching force: } \vec{F}_{stretch} = -k\,\vec{\delta}_{stretch}$$
$$\text{Spring bending force: } \vec{F}_{bend} = -k\,\vec{\delta}_{bend}$$
$$\text{Total spring force: } \vec{F}_{spring} = \vec{F}_{stretch} + \vec{F}_{bend} \quad (2)$$

Where: k= spring constant, $\vec{\delta}_{stretch}$= relative stretching of the spring, $\vec{\delta}_{bend}$= relative bending of the spring

6.2. Calibration of the Spring Manipulator

The spring manipulator is calibrated by the spring constant and by resetting the initial length of the spring. The rest position has a straight form. The spring manipulator has two ends: one is permanently connected to the stylus pen and the other end has a binding with the particle at the point of the ray intersection (see Figure 10).

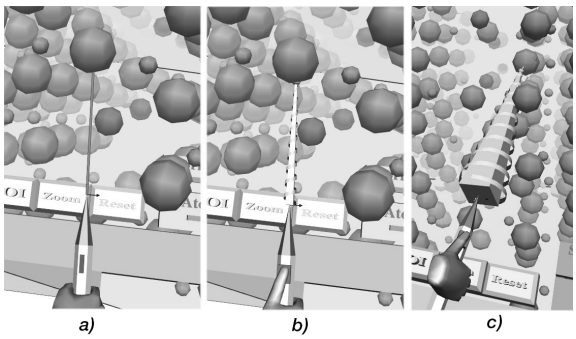

Figure 11: *Ray selection (a) and initial calibration of the spring manipulator (b) $k = 1eV/Å^2$ (c) $k = 5eV/Å^2$*

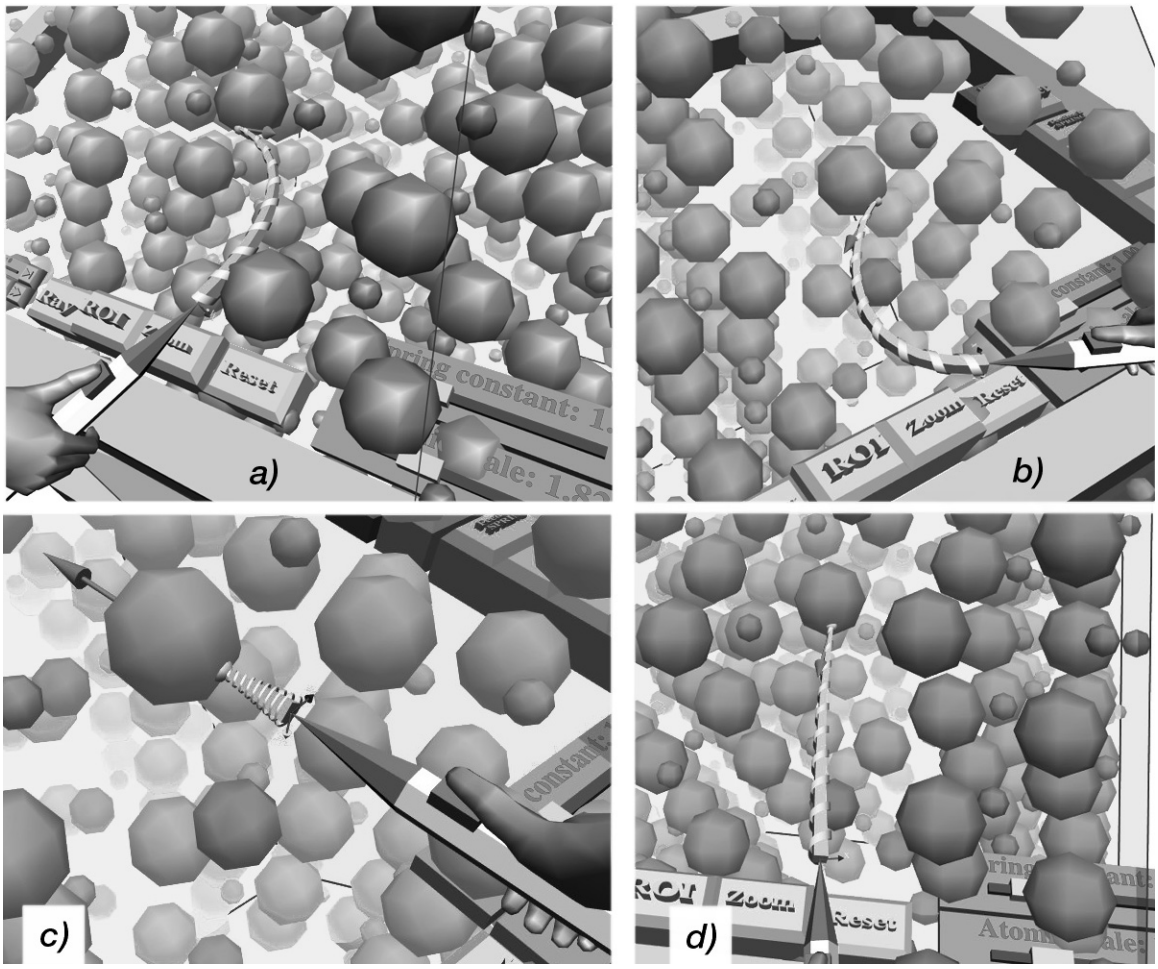

Figure 12: *Real-time particle steering using force feedback of the spring manipulator with spring constant* $k = 1eV/Å^2$

A moving particle in a running MD simulation can be selected with a ray technique, see Figure 11a. When the user clicks the button the initial spring length is computed, the ray disappears and the spring manipulator is displayed instead. The spring strength and dimensions are based on a given spring constant. Figure 11b shows $k = 1eV/Å^2$ and Figure 11c shows $k = 5eV/Å^2$. The user can change the spring constant with the slider (see Figures 12 and 13).

6.3. Particle Force Model of the MD Simulation

The original force model of the particle of DEMMPSI had to be adapted in order to enable the external user force. The force that the user exerts on a particle is also sent to the simulation and is added to the force delivered by the other particles due to long and short range interactions:

$$\vec{F}_{part} = \vec{F}_{short} + \vec{F}_{long} + \vec{F}_{user} \quad (3)$$

In the spring force feedback method we set $\vec{F}_{user} = \vec{F}_{spring}$. After the force calculation the particle positions are updated. Without taking precautions, the influence of the extra force to the simulation will finally result in a temperature change and a possible drift of the simulation which is caused because of a non-zero total momentum of the system. Therefore, after finishing the particle manipulation, the velocities of the particles are re-scaled in order to achieve the original temperature and to maintain zero momentum.

6.4. Examples of visual force feedback particle steering

In Figures 12 and 13, the spring manipulator is used to steer the particle in a running MD simulation. After particle selection and initial auto-calibration of the spring manipulator, the user can exert an external force on the particle through the visual spring mechanism into the force model of the MD simulation.

The arrow at the end of the spring shows the spring force which is computed from the spring deformations. Figures 12a and 12b show a user steering the Na^+ ion to the right or to the left respectively. The user is trying to find a new equilibrium position for the selected particle. Figures 12c and 12d demonstrate push and pull respectively.

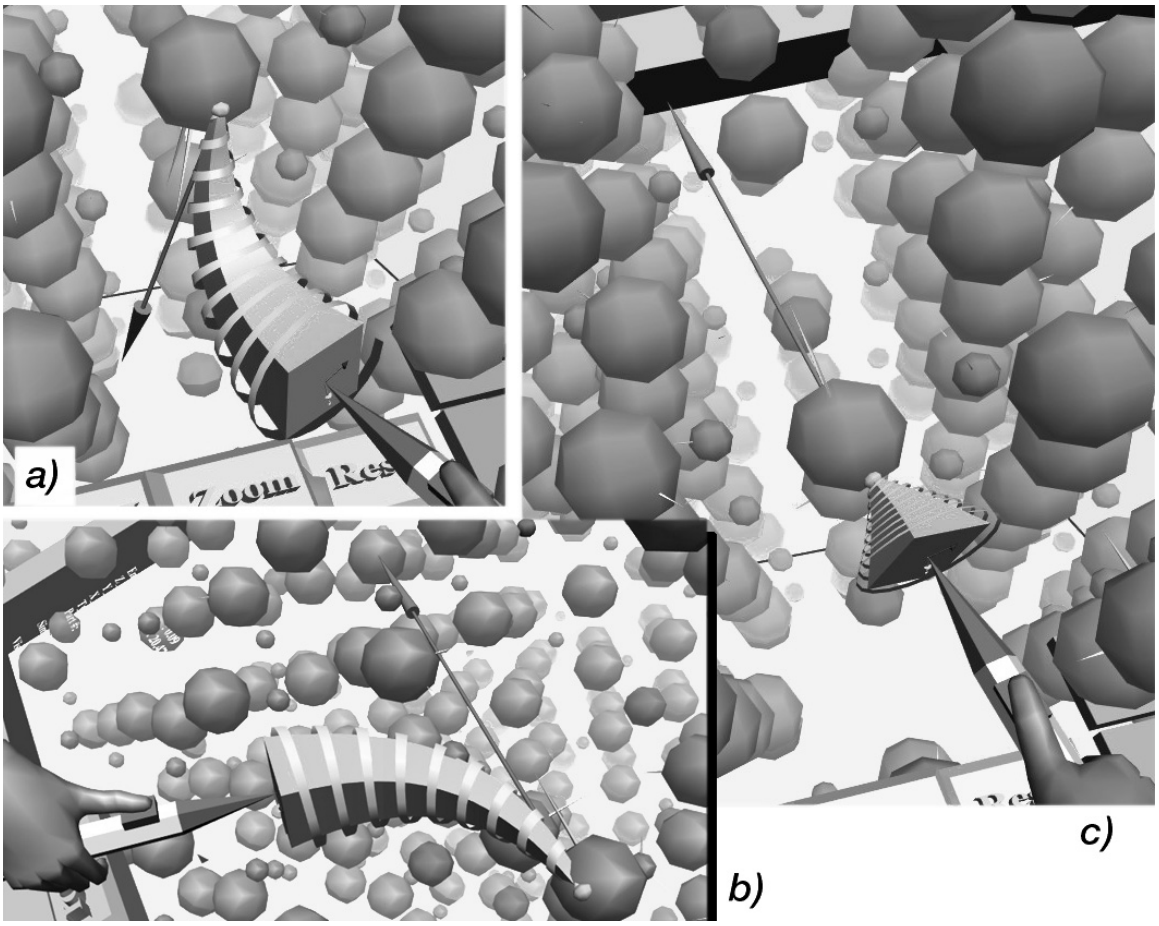

Figure 13: *Real-time particle steering using force feedback of a strong spring manipulator with spring constant* $k = 5eV/\AA^2$

In Figures 12 and 13, we can see a β-alumina system consisting of 2088 particles. There are atoms of sodium *Na* (red), oxygen *O* (yellow) and aluminum *Al* (small blue particles). The conduction plane is formed by the sodium ions. In Figure 13c the user looks down through this conduction plane.

In our visualization we use a depth cue technique based on distance-fog to improve the depth perception in a 3D visualization on the Responsive Workbench. It significantly improves orientation in the stereo images on the RWB (there we use a "black fog") as well as the RWB simulator images in this paper (here we have used a "white fog"). It helps a lot to distinguish between particles in the foreground and in the background.

Figures 13a,b,c show an *Na* atom being driven by a strong spring manipulator. When a weak spring is used, the spring force will be relatively small to steer the particle in the desired direction. With a strong spring is it possible to drag a particle even against strong atomic binding forces. In this particle visualization the user can also adjust the size of the displayed particles to see through the system. When the van der Waals atomic radius is used, the particle system is so dense that the user cannot look through it. Our MD simulation program provides the visualization client also with particle forces (green arrows) and velocities (white arrows), which are also visualized. Optionally, we can also interactively visualize derived grid data of the whole system, like kinetic energy, particle densities or particle potential, which shows important properties for particle steering (see Figures 17 and 18).

A user performing the particle steering in the MolDRIVE environment gets a complex visual feedback of his/her interactions with the particle.

An observation that we have made is that during particle manipulation the user normally introduces some local disturbance to the simulation. We have found that relaxation of the system takes about 10 to 30 fs (femto seconds).

With this paper we have also supplied some MPEG animations demonstrating our particle steering methods in real-time MD simulations [20].

7. Conclusions and Future Work

We have presented steering methods which enable users to interact with particles in real-time MD simulations on the RWB. The performance of the MolDRIVE system allows to steer in real-time systems with up to 4500 particles.

The virtual particle and spring feedback particle methods can be used to navigate a particle through relatively low energy barriers with little disturbance of the simulation. The visual spring helps users to keep contact with the steered particle and is experienced in a more natural way.

The virtual particle with the spring feedback is also a very useful technique especially when the user wants to check whether the particle could be placed at a given position rather than to be pushed to that position, which is the effect of the spring force feedback method.

Comparing the three steering methods, we have found that the spring force feedback method provides the smoothest way of steering the simulation to a desired configuration. Further, it seems that adding the force delivered by the spring manipulator to the particle makes sense regarding to the underlying physics. The user can visually experience the strength of inter-atomic forces and how much work it costs to reposition the atoms.

The researchers studying the β-Alumina electrolyte can now interactively perform experiments using the Responsive Workbench with the MolDRIVE system. The observations that are made in this virtual MD laboratory, when performing these experiments, create a better understanding of underlying processes. They also help to evaluate the simulation methods and certainly move this fundamental MD research further.

We have also reached our initial goal and all the particle steering tools are intuitive and very easy to use for non-VR experts. We have tested the spring force feedback also on steering of polymer simulations. A simple example of a carbon polymer is shown in Figure 16. Initial results indicate that the techniques can be easily extended to very different types of MD simulations.

In future, we want to use MolDRIVE to study protein simulations using the Gromacs [10] MD simulation program, where we would need to steer also particle (atomic) groups. Thus we will look for possible extension of our spring manipulators to move atomic groups.

Acknowledgements

We would like to thank other participating students, Jeroen den Hertog and Gerwin de Haan, for their contribution to the development and the implementation of MolDRIVE and helping with the pictures and animation for this presentation.

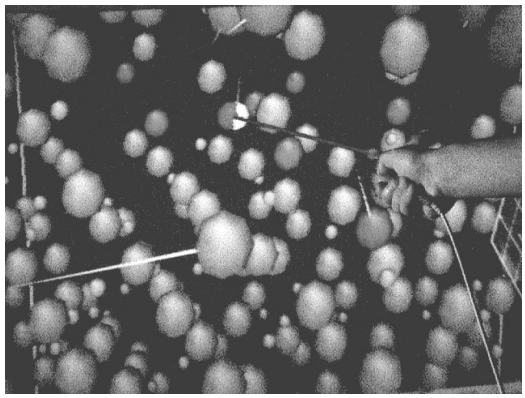

Figure 14: *RWB photo: user performing steering with virtual particle method.*

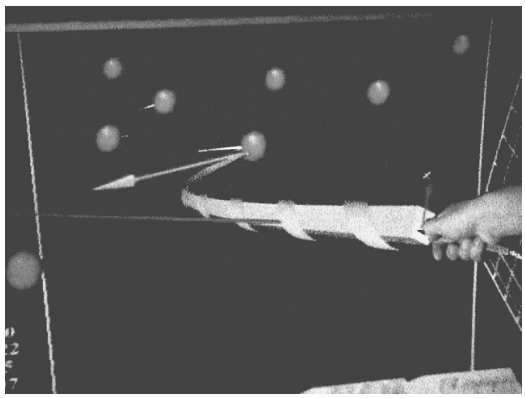

Figure 15: *RWB photo: the particle steering using the spring manipulator. The user moves the sodium atom through the conduction plane, the other non-sodium atoms are not displayed. The big arrow at the end of the spring manipulator shows the spring force and the little arrow shows the resulting velocity of this particle.*

Most of the figures in this paper were created in the RWB Simulator [4]. The real photos of the MolDRIVE system running on the Responsive Workbench are shown in Figures 14 and 15.

References

1. M. Koutek, F. H. Post, *Spring-Based Manipulation Tools for Virtual Environments*, Proc. of Immersive Projection Technology and Virtual Environments 2001, Springer, Stuttgart, Germany, pp. 61-70

2. M. Koutek, F. H. Post, *The Responsive Workbench Simulator: a Tool for Application Development and Analysis*, to be presented on the WSCG 2002, Pilsen, the Czech Republic, february, 2002.

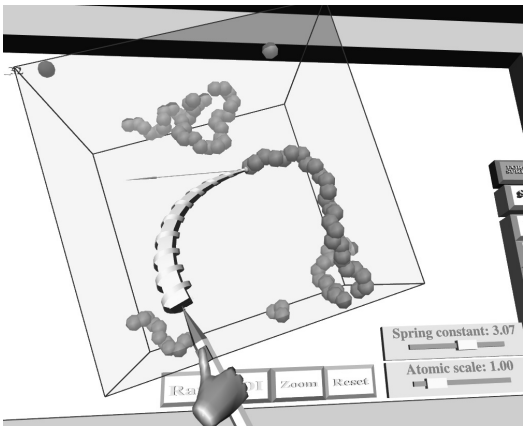

Figure 16: *RWB-sim: steering of a carbon polymer with the spring manipulator.*

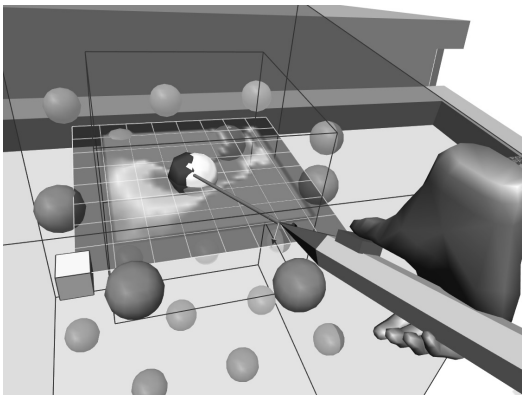

Figure 17: *RWB-sim: virtual particle steering method; a color data-slicer is used to display the potential around the particle.*

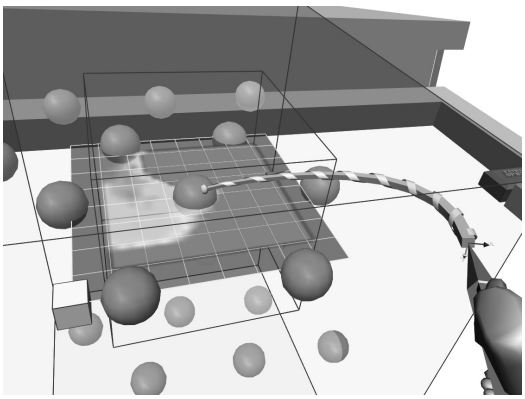

Figure 18: *RWB-sim: spring force feedback steering method; a color data-slicer is used to display the potential around the particle.*

3. M. Koutek, F. H. Post, *Dynamics in Interaction on the Responsive Workbench*, Proc. of Eurographics Virtual Environments, Springer, 2000, pp. 43-54
4. *The RWB Library and the RWB Simulator*, http://www.cg.its.tudelft.nl/~michal/RWBlib
5. J. Beckers, *Molecular Simulations of porous silica and* β*-Alumina*, PhD-thesis, Chapters 2, 5, Delft, 1999
6. J. Leech, J.F. Prins and J. Hermans, *SMD: Visual Steering of Molecular Dynamics for Protein Design*, IEEE Computational Science and Engineering, Vol.3,No.4:Winter 1996, pp.38-45
7. J.E. Stone, J. Gullingsrud and K. Schulten, *A System for Intercative Molecular Dynamics Simulation*, Proceedings of the 2001 Symposium on Interactive 3D Graphics, 2001.
8. B. Fröhlich, H. Tramberend, M. Agrawala, D. Baraff, *Physically-Based Manipulation on the Responsive Workbench*, Proc. of IEEE VR Conference, 2000, pp. 5-12
9. DEMMPSI MANUAL, *http://www.cp.tn.tudelft.nl/FAQ/MD/Main.html*
10. GROMACS, MD code in C/Fortran , freely availbale from: *http://www.gromacs.org*
11. NAMD, MD code in C++, freely availbale from: *http://www.ks.uiuc.edu/Research/namd/namd.html*
12. W. Humphrey, A. Dalke, K.Shulten, *VMD - Visual Molecular Dynamics*, Journal of Molecular Graphics 14(1):33-38, 1996.
13. J.F. Prins, J. Hermans, G. Mann, L.S. Nyland, M. Simons, *A Virtual Environment for Steered Molecular Dynamics*, Future Generation Computer Systems 15, Elsevier, 1999.
14. K. Arthur, T. Preston, R.M. Taylor II, F.P. Brooks Jr., M.C. Whitton, W.V. Wright, *Design and Building the PIT: a Head-Tracked Stereo Workspace for Two Users*, Proceedings of the Immersive Projection Technology Workshop, Ames, Iowa, May 11-12, 1998.
15. B. Grant, A. Helser, R.M. Taylor II, *Adding force to a Stereoscopic Head-Tracked Projection Display*, Proc. of IEEE VRAIS,1998.
16. R.M. Taylor II, *Scientific Applications of Force Feedback: Molecular Simulation and Microscope Control*, Course notes for Haptics, SIGGRAPH'99.
17. J.D. Brederson, M. Ikits, C.R. Johnson, C.D. Hansen, *The Visual Haptic Workbench*, Proceedings of the PHANTOM Users Group Workshop, October, 2000.
18. L.J.K. Durbeck, N.J. Macias, D.M. Weinstein, C.R. Johnson, J.M. Hollerbach, *SCIrun Haptic Display for Scientific Visualization*, Phantom Users Group Meeting, Dedham, MA, September 1998.
19. B. Grant, A. Helser, R.M. Taylor II, *Adding Force Display to a Stereoscopic Head-Tracked Projection Display* , Proc. of VRAIS' 98, Atlanta, Georgia, March, 1998.
20. *Animations of Particle Steering in MD Simulations:* http://www.cg.its.tudelft.nl/~michal/MDsteering

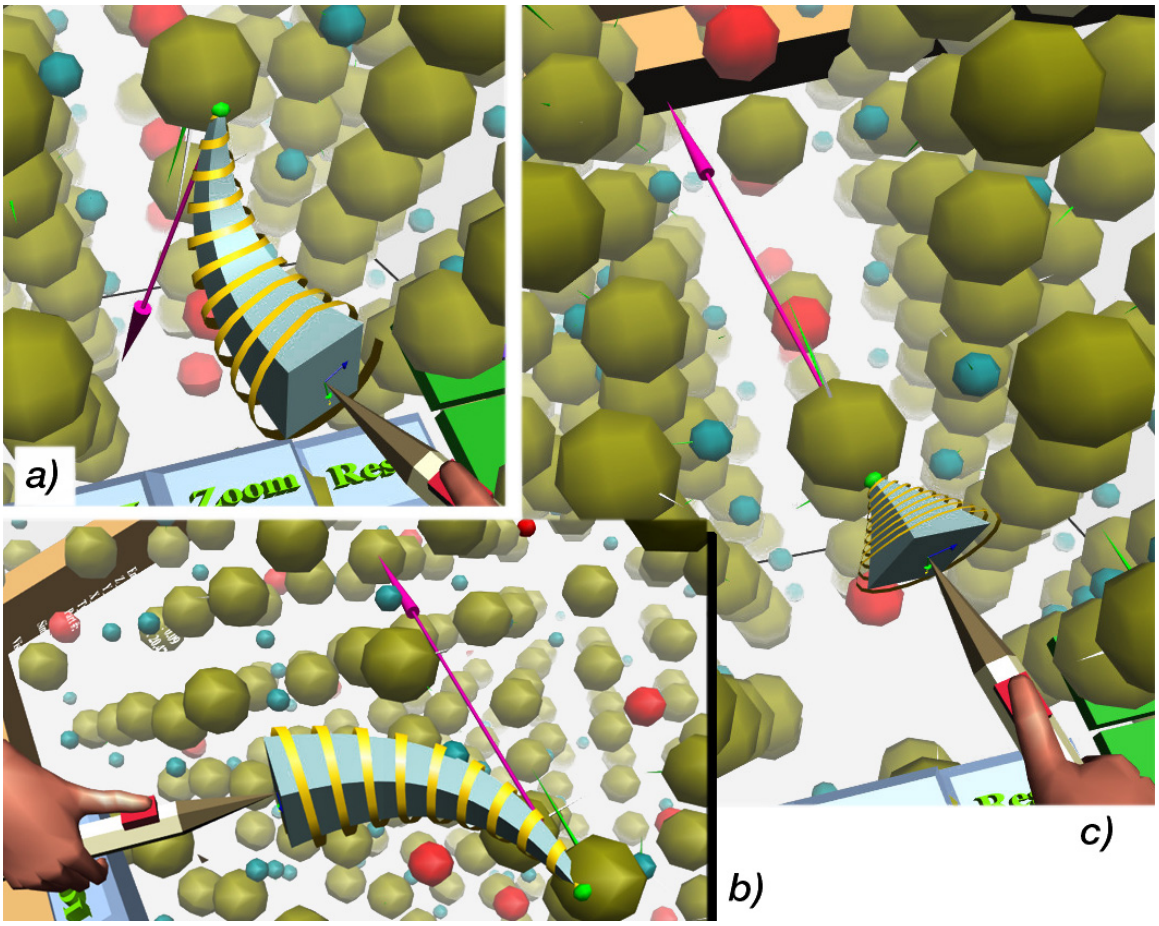

Figure 13: *Real-time particle steering using force feedback of a strong spring manipulator with spring constant* $k = 5eV/\AA^2$ *(the* β*-Alumina electrolyte system: 2088 particles)*

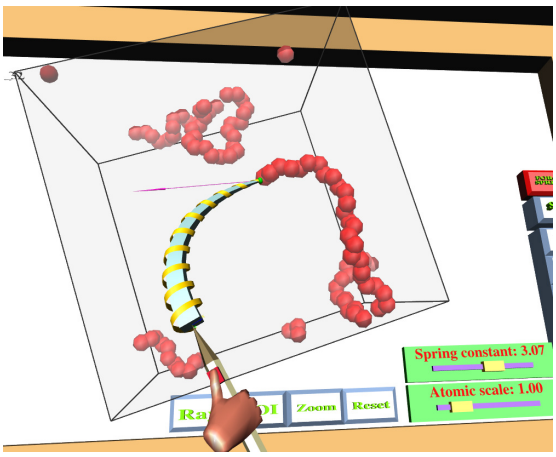

Fig. 16 RWB-sim: steering of a carbon polymer with the spring manipulator.

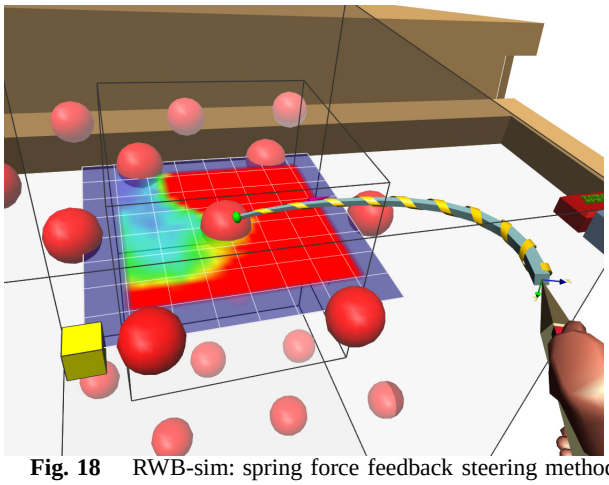

Fig. 18 RWB-sim: spring force feedback steering method; a color data-slicer is used to display the potential around the particle.