AC 2011-1500: DEVELOPMENT OF HAPTIC VIRTUAL REALITY GAMING ENVIRONMENTS FOR TEACHING NANOTECHNOLOGY

David Jackson, VCU Haptics Lab
Dianne T.V. Pawluk, Virginia Commonwealth University

Dianne Pawluk (PhD, Harvard) is an Assistant Professor of Biomedical Engineering at Virginia Commonwealth University. She teaches courses in the areas of computational methods, haptics and rehabilitation engineering. Her active research areas include haptic perceptual organization, the development of haptic assistive devices and methods for individuals who are blind or visually impaired, and the effective use of haptics in education. (Contact: dtpawluk@vcu.edu)

Dr. Curtis R. Taylor, University of Florida

Dr. Curtis Taylor, Ph.D. is an Assistant Professor in the Department of Mechanical and Aerospace Engineering at the University of Florida. His research focuses on understanding and developing new technologies (mechanical, electronic, optical, or biological) that utilize the unique capabilities of nanostructured/nanoscale materials. He works in both the Machine Tool Research Center and the Nanoscience Institute for Medical and Engineering Technology at UF. Before joining Florida, he was an Assistant Professor at the Virginia Commonwealth University (VCU) in Richmond, Virginia where he was Director of VCU’s NanoManufacturing (NanoMan) lab. He received his B.S. degree (1998) in mechanical engineering from the University of Maryland, and his M.S. (2002) and Ph.D. (2005) in electrical engineering and physics from the University of Arkansas. Before coming to Arkansas in 2000, he worked for one year as a software development project manager at Capital One Financial Corporation in Richmond, Virginia. Dr. Taylor has also held internship and research appointments with the U.S. Air Force, Central Intelligence Agency, United Technologies Corporation, and the National Center for Electron Microscopy at Lawrence Berkeley National Lab. Research interests include nanomanufacturing for the production of novel nanoelectronic and quantum devices, nanomechanical characterization of materials for development and improved reliability of nanodevices.
Development of Haptic Virtual Reality Gaming Environments for Teaching Nanotechnology

1. Introduction

Nanotechnology is a key high technology field that is becoming increasingly important to the United States’ economy. Maintaining leadership in key technologies, such as nanotechnology, is increasingly being recognized as important for American competitiveness. There is, therefore, a strong interest in attracting K-12 and undergraduate students to pursue future careers in this area. However, the abstract nature of current learning methods of how things interact and behave at the nanoscale (< 100 nm in any dimension) can be difficult for students to understand and conceptualize, thus leading to less interest in relevant career paths. One aspect that is fundamental to this conceptualization is how the forces between elements interact at this level. We are developing a hands-on teaching module for K-12 students about the forces that exist at the nanoscale and how they’re involved in nanofabrication. Rather than learning through abstract concepts, we intend for students to feel the forces during a virtual reality game to more concretely experience the concepts.

There are four important components of the module’s design for engaging students: the use of a virtual environment, the use of haptic feedback, creating a macro-world comparison to the nano-environment, and making the module into a game. For a virtual reality system (see Figure 1), its components usually contain visual and sound components, and can also contain a haptic component. Haptics is the field of perception that consists of the combination of the sense of kinesthesia (i.e., perception of forces and joint motion) and the sense of touch. Haptic technology usually involves force feedback (i.e., that produce controlled forces at the end point of the haptic device) and it is what is used here. The use of haptic feedback allows students to feel real, tangible forces in the ‘nano’ virtual environment, thereby allowing them to experience the invisible.

Figure 1: Block diagram of the virtual reality system.
Several different researchers have developed or used haptics for hands-on instruction of physics, engineering statics, dynamics and control systems in undergraduate curricula. For those that performed evaluations of the effectiveness of the haptic feedback, it was found that: 1) students were excited about the use of haptic feedback, and 2) the perceived value by the students was relatively high. However, in the only study to do a content evaluation, there was no difference between using a virtual simulation with only vision and one with vision plus haptics. It is possible that this may have been due to implementation issues and further investigation of using force feedback is warranted.

Many researchers (e.g., Sitti and his colleagues) have examined the use of scaled haptic feedback in real-time from an atomic force microscope (AFM) for research purposes. The haptic interface that is coupled to the AFM is used as a method of interacting with the nano-environment. Jones and her colleagues used a haptic device to actually control the AFM for teaching purposes. They reported results that suggested students found the experience engaging, developed more positive attitudes toward science and showed significant gains in their understanding of the targeted topic. However, the costs would be prohibitive for a classroom, the preparation would be difficult and the experience would be effectively limited to one student at a time – if they were allowed to use it at all (due to issues of costs and potential breakage). The other disadvantage is that the students had nothing to compare to the nano-world, particularly something they were already familiar with. Another expected key component of our learning module is the development of a macro-world for comparison so that students can relate their experience to what they will observe to be “normal” in the real world.

Finally, the module presents a gaming experience to further engage and excite students. Learning is more effective when it engages students’ attention and they are attracted to what is occurring for intrinsic rewards. Many K-12 students play digital games and, when presented with traditional teaching methods, feel disconnected from the classroom. They want learning experiences that parallel the exciting and engaging formats of digital games. The fact that people acquire new knowledge and complex skills from game playing, suggests its use in education. Positive features of games are that they are based on challenge, reward, learning through doing and guided discovery.

It should be noted that the goal of this research is not only focused on developing a teaching module, the primary concern of this paper, but also an evaluation of the four important components of the module’s design: the use of a virtual environment, the use of haptic feedback, comparing macro- and nano-worlds, and making the module into a game. For example, some results have shown that the use of haptics is not necessarily needed to teach some concepts. Digital games, although fun, may not transfer the content the students use for playing the game to new contexts in the real world.
2. Educational Concepts to be Taught

The two main concepts that we would like to communicate are: 1) an understanding of the dominant forces and their behavior as a function of scale, and 2) an understanding of how these dominant forces affect motion and the assembly of structures. These concepts are important to convey the challenges associated with nanoscale fabrication/assembly of structures and devices.

An important contribution to these concepts is that, as the size of matter decreases to less than 100 nm in dimension, the surface to volume ratio significantly increases (> 5 times). As a result, at the nanoscale, surface properties dominate volume properties and surface forces (e.g., van der Waals and electrostatic forces) can dominate gravitational (macroscale) forces. Figure 2 shows a comparison of scales for the macro and nano worlds.

**Figure 2:** Comparison of scales and associated changes in properties.

The concepts will be taught through the experience of building a structure (i.e. a house) at the macro and nano scales. At the macroscale, gravity is the dominating force, and one
that students will be most familiar. Students will have the opportunity to virtually touch, feel the weight, and inertial effects of handling building blocks (‘bricks’) to assemble a house. At the nanoscale, the dominant forces are the electrostatic and van der Waals forces. van der Waals forces are weak secondary atomic bonding forces encountered between two atoms or molecules, such as between molecular chains of amorphous polymers\textsuperscript{13}. The energy of the van der Waal’s potential is on the order of 1.6 eV and the force is $\sim 0.3$ nN (compared to $\sim$ N for the gravitational force at the macroscale). The van der Waals force is attractive in nature and presents significantly different behavior in motion and handling of objects at the nanoscale. Students will have the opportunity to interact with this force through virtually touching and assembling atoms into a structure—a ‘house’. The van der Waal force will be felt even before the student physically contacts the atom. Furthermore, once contact is made with an atom, the van der Waal and electrostatics forces prevent simply letting go of the atom.

Thus, students will understand how the motion of objects are affected by physical forces through the experience of building a structure (i.e. a house) at each scale. Students will be able to compare and contrast motion at the macro- and nano-scale. They will be able to explain that, unlike assembly in the macroscale, motions in the nanoscale are not simply reversible. Students will be able to describe how nanostructures are built, and to understand that all objects are composed of nanoscale ‘building blocks’ called atoms.

### 3. Choice of Teaching Environment

A virtual environment was chosen to enable us to rapidly create new environments easily at a low cost. It also facilitated active involvement by the user in controlling and modifying these experiences. For example, a main feature in the teaching module is the ability for the student to modify the physics to create a different environment. For example, gravity and inertia values could be manipulated or turned off. This gives the user a chance to experience a new law of physics that could not occur in the real world and to actively participate in creating these new worlds. It also allows for easy switching between experiencing one environment under a certain set of physical laws (e.g., the macroscale) and another set of physical laws (e.g., the nanoscale). The teaching module will utilize sight, sound, and force feedback to immerse the user into a learning experience.

The tools to achieve the immersion are OpenGL (Open Graphics Library) and Direct Sound for the visual and auditory components, respectively. OpenGL is an open source application interface for defining 2D or 3D graphics. Since OpenGL is a cross language, cross platform API it is virtually compatible with any home computer or laptop. Given that a classroom has enough haptic devices (see Figure 3) for the users, they could freely
and easily distribute the teaching module software, and begin the simulation immediately. For the auditory component, the current focus is on presenting sounds to deliver a realistic experience (e.g., collision sounds) but will be modified in the future to enable students who are visually impaired to navigate the environment as well (i.e., sounds will be used to tell the student how close they are to the nearest object).

Any force feedback device has the disadvantage (as compared to visual and auditory feedback) of requiring a non-standard interface device for the simulation: this will involve extra cost and installation. However, using a force feedback device will allow the concepts to be more tangible and intuitive. We are using a relatively inexpensive force feedback device, the NOVINT Falcon Force Feedback Game Controller (retail cost of ~$189) which has a freely available software development kit. The relatively low cost should make it more accessible to K-12 classrooms. The NOVINT Falcon device functions by detecting three-dimensional motion and producing vector forces in the linear directions; hence, it can touch and manipulate objects in the virtual environment, as well as feel the forces acting on it from the object. The Falcon will be the primary means by which to explore the environments in the teaching module. Students will use it to navigate and respond towards events in the simulation. The device allows for 3 degrees of freedom (DOF) (x, y, and z) for both motion and force. It also comes equipped with four buttons which can be programmed by the designer for specific interactions with the user.

Figure 3: NOVINT Falcon Force Feedback Device.
4. Algorithms and Implementation

We have created two different worlds in our virtual environment. In both of these worlds the concept is to build something, such as a house, out of the basic building blocks in that environment. For both environments, students will experience the forces involved as they move and interact with the world, as well as when they move the basic building blocks to form an object. In both environments, students will have the option to “turn off” one or more of the forces through the use of a menu structure. They can then selectively feel the contribution of each individual force. In addition, users will have the option to modify the properties of the “building blocks” such as their mass. This will also effect (or, in the case, of the nano-world, not effect) the interactions between the object and the student. This paper will primarily focus on the force feedback algorithms.

4.1 Macroscale

The Macroworld module is a simulation of building a house by stacking bricks. The two main concepts to teach the students are the force of gravity and inertia.

To start, the user is placed outside on a flat grassy field with a yellow line designated as the “loading brick zone” (left side of Figure 4). The user can move about in the virtual world by using the NOVINT Falcon, for which the position of the handle is represented by a small blue sphere. The small sphere acts as a cursor representing the user’s hand in the simulation. If the cursor is in front of the yellow line of the loading brick zone, then a brick will appear on the ground ready to be picked up. The user can now “pick up” a brick by pressing the center button on the Falcon device, but in order to hold the brick the user must hold the button down. If the user releases the button on the Falcon then the brick will drop to the ground. There are three buttons around the center button and pressing any one of those buttons will rotate the brick at a 90 degree angle to build walls that face another direction.
As the user lifts the brick, the Falcon device generates a downward force caused by the forces of gravity and inertia. The force of gravity is represented by the weight of the brick and exerts a constant downward force, whether the brick is being moved or held in place. The force of gravity continues to be exerted on the brick even when it is released from the user’s hand. This will allow the brick to fall to the ground or on top of other bricks. Importantly, if the brick is placed on the edge of another brick or a brick wall with enough of its center of mass over the edge, it can tip over and fall.

Inertia is the resistance projected from a change of velocity to an object. Thus, it does not exist if the brick is held constant or moved at a constant velocity. However, if the user moves the brick in one direction than jerks the object the opposite direction, the inertia of the object will go against the user while the velocity is changing. In modeling of the force feedback, the brick’s inertia consists of the negative acceleration of the cursor multiplied by the mass. The cursor acceleration is derived from a record of the last five positions of the cursor.

With,

- \(a\) – cursor acceleration vector
- \(B_i\) – the set of previous positions with index 4 being the current position

\[ a = B_4 - 2B_2 + B_0 \]

and with,

- \(M\) – the brick’s mass
- \(F\) – the inertia force generated by the Falcon device

\[ F = -Ma \]
If the brick being held is released while the cursor is moving, the brick’s velocity will inherit the cursor’s velocity at the point of release, or the tangent of the cursor’s motion at the release point (see Figure 5). At that point, the brick’s velocity will not deviate with only the exception of the downward force of gravity.

**Figure 5:** Diagram of brick and cursor motion where \( v \) is the brick’s velocity vector and \( R \) is the point where the brick is released from the user.

The trajectory of the brick’s motion is modeled inside of an iterative loop. The velocity vector and the acceleration due to gravity are always added to the brick’s position on each successive loop for as long as the program is running. If the brick collides with another object the velocity is changed, and due to the nature of the brick’s inelasticity, the velocity’s magnitude eventually reduces to 0.

With,

- \( B \) – brick position
- \( v \) – brick’s velocity vector
- \( g \) – gravity force
- \( t \) - the time step

**WHILE** program is still running (keep looping)

IF collision occurs THEN change \( v \) according to the type of collision

\[
\begin{align*}
  v &= v_{\text{previous}} + gt \\
  B &= 0.5gt^2 + v_{\text{previous}}t + B_{\text{previous}}
\end{align*}
\]

IF \( B.y < \text{ground level} \) THEN \( B.y = \text{ground level} \)

If there are already bricks on the ground, the algorithm will check to make sure that the moving brick does not fall through the already present bricks. If the brick is thrown a considerable distance, it will bounce due to its coefficient of restitution. However, after each successive bounce the brick’s velocity and height of bounce will start to diminish considerably due to the fact that the brick is far from perfectly elastic.

There are also more forces and collision at play then simply the cursor (i.e., the hand) interacting with the brick the user is holding, or the brick with the ground. The other bricks are also surfaces that cannot be penetrated. Collisions are enabled so the bricks
cannot move through each other: a held brick can be stacked on top of the grass or on other bricks, but cannot fall through the other bricks or the ground. If a brick is resting on another brick tilted at an angle, the static frictional force could keep the brick from sliding down depending on the height of the slope.

4.1.1 Implementations of Collisions

Since the virtual environment is drawn in 3D graphics, the implementation components handling collisions required a solid understanding of geometric vector math. The two main components in the software that handles this are the collision detection and the collision response algorithms.14

Collision Detection

In navigating the cursor through space, it was important for it not to pass through any bricks or the ground. The problem that needed to be solved was to find a set of instructions to determine if the cursor position intersected any object surfaces, and, if so, then what object was impacted and where was the point of intersection. Once these questions have been answered then there remains to properly respond to the collision.

The collision detection algorithm assumes that the cursor is represented by an infinitely small point. The collision detection determines if the cursor point penetrates any of one of six sides of a brick (see Figure 6). The necessary information to begin examining a collision is the cursor’s position, cursor’s velocity, and the four corners of a brick’s side.

With, \( P_0 \) – cursor’s previous position
\( v \) – cursor velocity \((P_1 - P_0) / t\)
\( P_1 \) – cursor’s position, found by the equation \((P_0 + v^t)\)
\( A, B, C, D \) – the four corner points of a brick’s rectangular side
\( n \) – the surface normal to the plane of the brick’s side
\( s \) – the distance between the cursor’s previous position and intersection point when moved along the direction of \( v \)

If \( t = 1 \),
Figure 6: Diagram of collision detection algorithm.

The collision detection algorithm has to locate the plane intersection point where the brick’s surface exists and to determine if the intersection point is within the four corners of the brick’s side. The plane intersection point is where the cursor would intersect on the plane of a brick’s side if the cursor’s velocity vector was sufficiently long enough to reach the plane. Whether the cursor intersects the brick’s surface depends largely on knowing the plane intersection point.

The plane intersection point can be found by adding a sufficiently long vector to the previous cursor position, $P_0$, that points in the same direction as the cursor’s velocity. The problem is we don’t know how long this vector needs to be, yet. On the other hand the direction of this vector can be obtained by normalizing the cursor’s velocity to get $u$. So we know that the intersection point, $I$, is where:

$$ I = P_0 + s \cdot u $$

Let $s$ be the distance between the cursor’s previous position and the plane intersection point when moved along the velocity vector. This can be found by the following:

$$ s = (n \cdot w) / (n \cdot u) $$

where, $w$ is the vector difference of the cursor’s previous position and the origin, an arbitrary point on the plane. Point $A$ will be the chosen origin for this scenario.

$$ w = A - P_0 $$
The plane’s normal, \( \mathbf{n} \), can be found by arbitrarily choosing an origin and subtracting it from any other two points that exist on the plane to obtain two vectors, \( \mathbf{a} \) and \( \mathbf{b} \) (see Figure 6).

\[
\mathbf{a} = \mathbf{B} - \mathbf{A} \\
\mathbf{b} = \mathbf{D} - \mathbf{A} \\
\mathbf{n} = \mathbf{a} \times \mathbf{b} \quad \Rightarrow \text{then normalize}
\]

All the information required to solve for the plane intersection point are now known. However, the assumption cannot be made that the moving cursor will actually intersect the brick’s surface. If four lines from each corner of the brick’s side are connected to the plane intersection point and the sum of those angles equals 360 degrees then the plane intersection point is inside the brick’s rectangular side (see Figure 7). Otherwise, it is not.

**Figure 7:** Diagram showing plane intersection point possibilities, where \( \mathbf{I} \) is the plane intersection point and \( \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4 \) are the four vectors between the corner points and the plane intersection point

In reference to Figure 7, we need to find the acute angles between the points to see if they sum to 360. First, we need to find the vectors that connect to the plane intersection point to the four corner points, then normalize the set of vectors \( \mathbf{v}_i \) to produce unit vectors \( \mathbf{u}_i \). We then take the arc cosine to determine the acute angle between two vectors. If the sum of all the angles is equal to 360 degrees, then the plane intersection point is inside the brick’s side. If the sum is less than 360 degrees then no collision will occur on that brick’s side and the collision detection algorithm is exited.
The collision detection of the cursor colliding with a brick’s side is complete, but that was only one out of six sides. The problem that remains is what if the cursor’s velocity vector penetrates multiple brick surfaces (e.g. the front and bottom). The surface collision detection algorithm must be repeated for each side of the brick. The collision with the shortest distance found among any other possible collisions is interpreted as the actual collision.

Collision Response

The macroscale physics was created to mirror the real world physics, so just like a hand would not be able to go through a brick, neither can the cursor. Collisions involve forces that change an object’s velocity. To simulate collisions, the basic idea is to model all object surfaces as a spring, including hard objects like bricks. The spring compression equation that is used is given below, with k being the spring constant and x being the distance compressed.

\[ F = -kx \]

The distance compressed is described by the distance normal to the surface that is penetrated by the cursor. Since the Falcon Device is a 3D touch device, the force projected does not equate to a single linear value. The force feedback is in the form of a vector, with direction and magnitude, and the spring constant.

\[ F = -k(x, y, z) \]

If a collision is detected then the force feedback occurs using some of the information gathered from the collision detection algorithm (described above). In order to do this, the shortest distance between the cursor and the object’s surface needs to be found as it is the penetration distance. To find the shortest distance from the cursor’s position \( P_1 \) and the brick surface normal \( n \) (see Figure 6). Once the distance is known, the force can be calculated.

\[ w = P_1 - A \]
\[ d = w \cdot n \]
\[ F = -kd n \]

Modifying Forces

The macroscale simulation will give the user the option to modify the physics of the virtual environment. The values of gravity, the coefficient of restitution, frictional forces, and inertia can be manipulated or turned off. Modifying a particular force and isolating the effects allows the force to be more observable by the student. In addition, the mass of the bricks can be modified, which will affect all forces.
4.2 Nanoscale

The nanoworld module is a simulation of building a “house” by stacking atoms. The main concept to teach the students is about van der Waals forces including the flick to contact instability and the approach-retract hysteresis.

The virtual environment represents what can be seen and felt at the atomic force microscopic level (see Figure 8). The surface floor is a layer of teal atoms that cannot be moved due to powerful adhesion forces between them. A different group of orange atoms will rest on top of the layer of teal atoms. The adhesive forces of the orange atoms are strongest to like atoms, and stronger for the AFM than the teal atoms. The user will be navigating an AFM cantilever tip in the nano-world to create a “house” with the orange atoms.

![Figure 8: Screenshots of Nano-world Simulation.](image)

Similar to the macro-scale simulation, the user is placed in an environment with teal atoms representing a flat surface below the user. A yellow line will again designate the “loading atom zone”. If the cursor with the AFM tip is moved in front of the yellow line of the loading atom zone, then an atom will appear on the ground ready to be picked up. The user can now “pick up” the atom by moving toward it. As the cantilever tip approaches the orange atom, the user will feel the attractive van der Waal’s force before the cursor reaches the atom (refer to Figure 9). Although, for the attraction between two atoms, this is normally approximates by the following equation:

\[ z = \text{distance between the AFM tip and the atom surface} \]
\[ F = \text{force generated} \]
\[ A = \text{constant} \]
\[ F = \frac{A}{z^6} \]

We will use the following equation as it makes the van der Waal’s forces more apparent:

\[ F = \frac{A}{z^2} \]

**Figure 9:** A is the cantilever tip, B is the orange atom, C is the closest point between the atom and the cantilever tip. The distance between A and C are used to generate the van der Waals forces. \( w_1 \) signifies the distance from the orange atom the tip needs to be to activate the van der Waals forces.

If the user continues to move the cantilever tip closer to the atom, the user will experience the flick instability. Because the orange atom is more strongly attracted to the cantilever contact than the surface atoms, it will be the component that moves. Therefore, when the distance between the cantilever tip and the orange atom gets within a certain distance (the snap to contact radius), the atom “moves” instantaneously to attach itself to the tip. Note that no buttons are involved in picking up the atom in contrast to the macro scale. Once the atom adheres to the cantilever tip, it is programmed to move as part of the tip structure and can be moved freely in the virtual environment. The orange atom is not attracted back to the teal atoms, as its attachment to the cantilever is stronger than to the teal atoms.

The teal atoms can also attract the cantilever, but their bonds to each other are much stronger than to the cantilever. So it is the cantilever which snaps into contact with them.
When the cantilever tip is within a certain radius of the teal atoms, it will experience the van der Waals forces provided by all the atoms on the floor. This will be the vector summation of all the van der Waals forces acting on it. When the cantilever gets within a certain radius from one of the teal atoms, it will be the component that moves into contact with the teal atoms. Once in contact with the atom it will feel adhesion forces to it. Both the flick to contact and the adhesion forces are modeled by the following equation:

\[ k = \text{strength of the force} \]
\[ z = \text{distance between the AFM tip and the atom surface} \]
\[ F = kz \]

Once the flick to contact occurs, the AFM tip is programmed to continue to feel the adhesion forces until a force ten times that of the flick to contact force. At this point, the AFM will be suddenly released from its attraction to the teal atoms and free to move in the simulation.

If the AFM cantilever tip “picks up” an orange atom, it is free to move around the virtual space with the atom attached (see Figure 10). If the attached orange atom comes in contact with another orange atom (by moving inside the range of the atom’s van der Waals radius) then the van der Waals force pulls the atom and the cantilever tip to which it is attached, towards the other orange atom. The force that is exerted is the same equation as given above for the van der Waals forces, but this time it is between the two surfaces of the atom. If the two atoms come within a certain distance of each other, the flick to contact/adhesion force equation is activated. What is different now is that the orange atom has a much stronger attraction to the other orange atom than the cantilever tip. If the user then tries to move the cantilever tip away, there will be a point, again, approximately ten times the flick to contact force, at which the cantilever tip breaks away from the atom, leaving the two atoms attached to each other.
Figure 10: Atom B is being dislodged from the AFM cantilever tip by greater adhesive forces from atom C. \( w_1 \) and \( w_2 \) indicate the radii at which the van der Waals forces are activated.

The collision detection primarily checks if the AFM cantilever tip or the atoms breach an adjacent atom’s Van der Waals radius. When this occurs the Van der Waals attractive force activates and the Falcon device mimics the force.

Modifying Forces

The nanoscale simulation gives the user the option to modify the physics of the virtual environment just like in the macroscale simulation. The user has the option to change the attractive forces between the different types of atoms and between the atom types and the cantilever tip. In addition, the mass of the atoms can be changed, as in the macro simulation for the mass of the bricks.

5. Preliminary Results and Discussion

As the simulations are still in the development and testing stage to validate the program works, and the story line has yet to be developed, we have not yet evaluated our teaching module for its effectiveness. However, we have demoed our teaching module to students and faculty. The reaction to the simulation has been very positive. The students were very much engaged in the simulations and even graduate students familiar with haptic feedback were excited by the forces they could feel through the NOVINT Falcon. The most unexpected components that excited students were the ability to throw a brick and to try and throw an atom. This was certainly not a bad thing as in both cases important
concepts are taught: namely gravity and inertia for the macro scale simulation and adhesion forces for the nanoscale.

After validation of the computer program, we will develop a storyline and a set of questions to lead K-12 students through the simulation. We will then look at evaluating the use of our two simulation games on: (a) improved learning and retention of nanotechnology concepts, and (b) an increase interest in STEM fields (particularly nanotechnology) by these same students. As a control in studying the impact of using haptics on learning, we will compare the outcomes with the haptic learning modules with ones using other techniques: (a) vision only, (b) vision + audition, and traditional teaching methods.

Two different assessment tools will be used. First, during the use of the teaching modules, field notes will be taken about the questions students ask (beyond how to use the Falcon), analogies used when describing the worlds to the experimenters, as well as affective words and haptic words. In addition, there will be a pre-assessment and post-assessment with all learning groups. The pre-assessment will consist of two components: 1) an evaluation of the student’s knowledge about nanotechnology and 2) an evaluations of their interest in nanotechnology and STEM fields. The post-assessment will contain the same questions as the pre-assessment with some additional questions. Among the additional questions will be ones that will examine the student’s ability to transfer their knowledge to new but relation situations such as drug delivery. We expect the use of a virtual haptic environment combined with gaming will be an effective method for conveying knowledge about nanotechnology and generating excitement about the field.

6. Acknowledgements

This work was supported by NSF EEC grant #0934822 to D. Pawluk and C. Taylor.

References


