Lock Picking Simulation Using Visual and Multi-Haptic Displays

Karl Arthur & Courtney Parsons

University of Utah

ABSTRACT

Lock picking is one of few activities that can be performed using only haptic feedback. Because lock-picking feedback is limited to the sense of touch, and the inner workings of locks are intentionally hidden to prevent easy inspection, the amateur lock-picker is constrained to gain experience and a sense of the internal workings through touch alone. The technique used to compromise a simple pin tumbler lock relies on haptic feedback experienced through both hands, with one hand managing a tension wrench and the other hand manipulating a pick. By providing certain cues through two haptic displays the sensations of lock picking can be mimicked, allowing the user to experience what it feels like to pick a lock. By providing a working visual model, the inner workings of a pin tumbler lock are revealed and allow the user to gain a visual sense (coupled with the haptic sense) for how this lock can be compromised. This haptic/visual model, if generalized, could be used to model any number of complex locks on the market, enabling locksmiths and lock-designers to test the security of various locks.

KEYWORDS: Haptic, visual, virtual, multi-display, simulation, lock, key, wrench, pick, feedback, locksmith, design.

INDEX TERMS: K.6.1 [Management of Computing and Information Systems]: Project and People Management—Life Cycle; K.7.m [The Computing Profession]: Miscellaneous—Ethics

1 INTRODUCTION

Pin tumbler locks are common in the society in which we live, being used in both deadbolts and doorknobs. We rely heavily upon these simple locks for the protection of our homes, yet an experienced lock-picker could easily compromise one of these locks in less than sixty seconds.

The main purpose of this simulation is to mimic the internal dynamics of a simple pin tumbler lock, consisting of complex interactions. As locks become increasingly complex and expensive it may become impractical for locksmiths to own every lock available on the market. In practicing for their trade, it may also be useful for them to see the parts move inside the lock as they experience the haptic sensations. Then, when necessary, the locksmiths would be prepared to compromise the actual locks.

Another motivation for a lock picking simulation is to provide lock manufacturers with the ability to test the security of new lock designs within an interactive environment. If the program were generalized to handle whatever lock model is uploaded, then the lock-designer could immediately (and cheaply) find obvious flaws before producing actual parts and locks.

University of Utah
Department of Mechanical Engineering
50 S. Central Campus Drive, MEB

The authors of this article learned to recognize the haptic cues needed to pick pin tumbler locks from materials written by the master locksmith, Steven Hampton [3 & 4]. In his works he described what to feel for when picking various locks, suggests various methods for improving ones finger haptic sense, provides images to help conceptually pick various locks, and also provides caution against illegal use of his techniques. Many books and sources are available for learning how to pick various simple locks, but few refer to these techniques as belonging to the haptics field of study (e.g., see [1, 3, & 4]).

2 THEORY

To understand the theory of lock picking it is necessary to understand the internal workings of the lock and the tools used to pick locks. Figure 1-1 shows a cross-sectional view of a pin tumbler lock, and will be referred to throughout this paper.

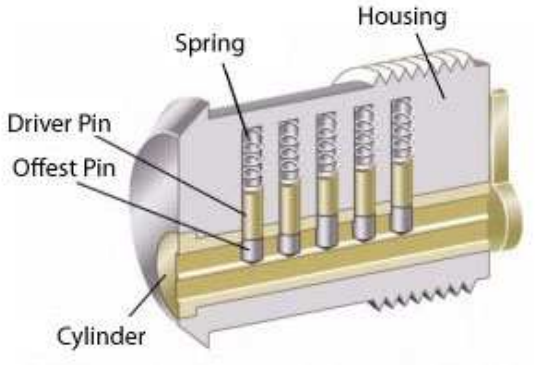

Figure 1: Cross-sectional view of a pin tumbler lock [11]

2.1 Pin Tumbler Locks

A typical pin tumbler lock consists of the housing, five or six sets of pins and springs, and a cylinder (shown in **Error! Reference source not found.**). The housing is secured to a door as either a deadbolt or doorknob. It keeps all of the internal parts aligned and contained.

The cylinder fits tightly within the housing and is where a key is inserted. When the proper key is inserted, the cylinder can rotate freely inside the housing. The cylinder is secured against the housing at the back (opposite the key entryway) so that when the proper key is inserted the cylinder cannot be removed from the housing.

Vertically bored into the top of both the housing and the cylinder is a hole for each set of pins. Each pin set consists of a spring, driver pin, and any number of offset pins. The spring gives the pins a inward force to push the pins against the key when inserted. All driver pins within a lock are the same length, and placed between the spring and offset pins. The lengths of the offset pins vary to match the displacement grooves of the key. When the key intended for this lock is inserted, the offset pins all align with the shear-line separating the housing from the cylinder. In the case of

locks that accept master keys (such as are common within apartment buildings), multiple offset pins are used, allowing more than one key to match the offset pins with the shear-line. (Consequently locks that accept multiple keys are inherently easier to pick because more opportunities exist to properly align the offset pins.) If an incorrect key is inserted into the lock the pins will not align with the shear-line and thus the cylinder will be prevented from rotating within the housing.

2.2 Picking a Pin Tumbler Lock

The Picking of a pin tumbler lock can be accomplished using the two instruments shown in figure 2. The first (shown on the left) acts as a tension wrench, and the second (shown on the right) acts as the pick (modeled here using a bent safety pin). The tension wrench is inserted into the base of the key entryway and then a small force is applied to the wrench. This small force places the offset pins in shear which provides sufficient frictional force to overcome the spring force and hold the pins in position.

At the same time that the wrench is holding the pins in place, the pick is inserted into the key entryway and used to apply a force opposite the spring force, pushing the pins outward from the center of the lock to align the pin separation (i.e., the separation between the driver and offset pins) with the shear-line (i.e., the separation between the housing and cylinder). When positioning the pins the inward spring force must be overcome at the same time as the frictional force provided by the tension wrench. This involves a delicate balance in determining how much force to apply both to the wrench and the pick. If too little tension is applied then previously compromised pins will come out of alignment, but if too much tension is used the pins cannot be placed and may possibly damage the lock.

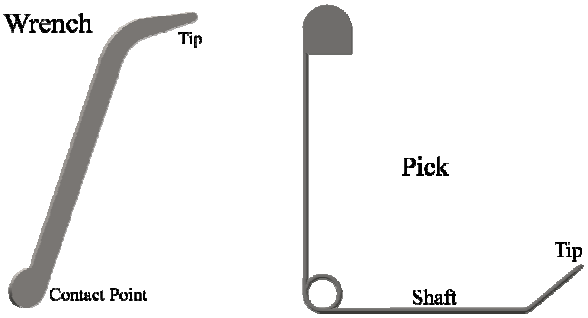

Figure 2: Picking instruments wrench (left) and pick (right) used in the lock picking simulation.

Some locks provide additional measures of protection against picking. One of these measures includes replacing the cylindrical driver pins with mushroom pins, shown in figure 3. As the mushroom pin reaches the shear-line it allows the cylinder space to move slightly, providing the lock-picker with a false sensation of compromising the pin. Another security measure taken may include spring-loaded cylinders. These make it difficult to provide the proper amount of force to overcome the rotational spring before matching the delicate force balance needed to place the pins. Mushroom pins were not considered in this simulation, but spring-loaded cylinders were.

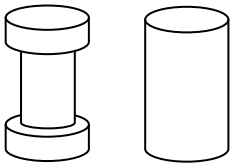

Figure 3: Mushroom (left) and cylindrical (right) driver pins.

3 RESULTS & DISCUSSION

Using C++ (a programming language), OpenGL (graphics library application programming interface), and Chai3d (graphic and haptic display libraries), a working lock picking simulation was created to mimic the picking environment and corresponding haptic sensations felt while picking a pin tumbler lock. As shown in figure 4 a computer monitor was set up to visually display the virtual lock picking environment, and two haptic devices were used to display the haptic feedback to the user. Using keystrokes the operators of the simulation are allowed to change the camera view, (spherically moving left, right, up, down, inward, and outward), and remove or make transparent the housing and cylinder parts of the lock model in order to see the pin-pick interactions. The two haptic displays consist of a haptic paddle (left) to display the wrench sensations and the PHANToM Omni (right) to display the picking sensations.

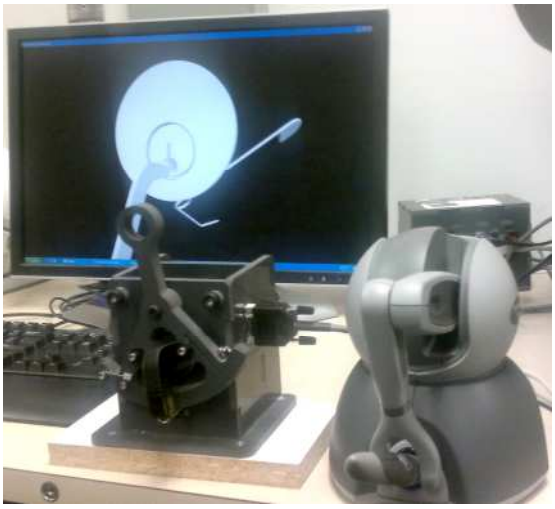

Figure 4: Image of simulation setup, computer monitor display with keyboard, haptic paddle (left), and PHANToM Omni (right).

3.1 Haptic Devices

As mentioned in section 2.2, both hands are used when picking a lock. One hand applies the small shear force to the pins using the tension wrench and the other hand manipulates the pick to position the pins. Because these two tasks can be accomplished independent of one another it was simple to simulate the haptics separately, with one device used for simulating the tension wrench sensations and another device for simulating the picking sensations.

In developing the simulation for two separate haptic devices no custom device needed to be built. However, since both of these devices are relatively large (compared to the actual lock) the user's hands would be placed farther apart relative to each other than they would when picking a real lock. Also the program was written assuming that no head-mounted display would be used, thus also placing disparity between the visual and haptic positions of the lock in space. When the authors received user feedback it seemed that the added distance between the hands did not appear to alter the sensations of the simulation. This is to be expected since lock-picking is usually done without visual feedback and with mostly-independent haptic sensations. (Some sensations, such as the compromise of a pin, can be felt with both hands.)

3.1.1 Pick Haptic Interface

The PHANToM Omni was chosen as the haptic interface for the pick. It is a commercially available device that has three degrees of freedom for sensing position, three for sensing orientation, and three for providing force feedback, (shown on the right in Figure 4). This device was chosen for this simulation because the long and slender haptic interface tool would be held in a similar manner as an actual pick, thus further adding to the immersion of the simulation.

Because the PHANToM Omni senses with six degrees of freedom but provides forces in only three degrees of freedom this device is considered a partial-force feedback system [9 & 10]. In order to provide the most realistic picking sensations, (ignoring the bulk of the device compared to an actual pick), and mimic the interactions within a small constrained space in a virtual environment one must use a full-force feedback system. However, since most forces in actual lock picking occur within a small space, and since a partial-force feedback system provides a more economical approach to simulating this environment than full-force feedback, the authors chose to limit the simulation to partial feedback with the option to constrain the user and pick to the plane of the pins. Thus, this simulation uses a single small sphere placed at the very tip of the pick to detect interactions with the lock environment, and all forces are spatial (i.e., no torsion involved). In the end it seemed that most users automatically oriented the pick anyway.

3.1.2 Wrench Haptic Interface

Although a tension wrench can technically move through space with six degrees of freedom, when picking a lock the cylinder on which it provides torsion constrains the wrench to rotate about a single axis (i.e., that of the cylinder's axis of rotation). A haptic paddle device was chosen as the haptic interface for the tension wrench because it also operates with a single degree of freedom, which also happens to be rotational. The haptic paddle used for the demonstration was manufactured at the University of Utah.

In order to more closely mimic the actual motion of a tension wrench for the purposes of this simulation it was undesirable to have the user manipulate the handle on the top of the device, or to push on the angled side of the paddle. The force applied to an actual tension wrench occurs as far away from the lock as reasonably possible to improve sensitivity to the vibrations within the lock. Therefore, the desired location for applying the force should be located at base of the handle. An additional fixture was designed and attached to the paddle so that the user could apply force in the location that most closely mimicked an actual tension wrench as shown in figures 4 and 5.

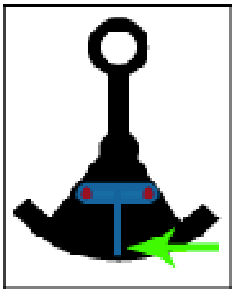

Figure 4: Haptic paddle attachment (blue) and force (green).

3.2 Keyboard Interactions

Keyboard inputs were added to allow the user to make modifications to the view and improve their experience with the simulation. The following is a list of the important keystrokes and the effect each has on the simulation:

Key	-	Action
w	-	Toggles between solid and **W**ireframe views for housing, cylinder, and wrench
h	-	**H**ides the lock housing and cylinder so as not to be seen or felt
p	-	Sets **P**ick as interactive tool; disables key
k	-	Sets **K**ey as interactive tool; disables pick
m	-	Aligns the tool to the plane of the pins (at **M**iddle)
c	-	Simulates the sensation of **C**ompromising a pin on haptic paddle; causes the paddle to move slightly
i	-	Moves the camera **I**nward towards lock
o	-	Moves the camera **O**utwards away from lock
r	-	**R**esets or initializes Omni encoders
←	-	Moves camera view to the **left** in a cylindrical manner
→	-	Moves camera view to the **right** in a cylindrical manner
↑	-	Moves camera view **up** in a spherical manner
↓	-	Moves camera **down** in a spherical manner
ESC / x	-	E**x**its the program

3.2.1 Viewing the Environment

The lock-picking environment can be viewed from any angle by using the arrow keys on the keyboard. The view will change in a spherical coordinate system always pointing to the same point within the lock. This option is especially useful when a user wants to see the internal parts of the lock as they pick. Figure 5 shows some views of the tumbler lock and also depict various keyboard options.

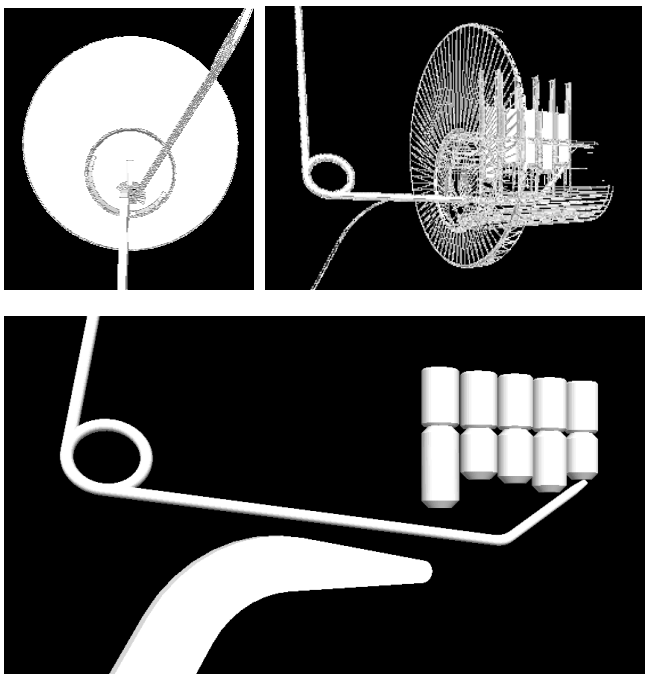

Figure 5: User modified views: front (top left), wireframe (top right), and hidden (bottom).

3.2.2 Pin Interactions

The workspace of the PHANToM Omni ranges from 160mm in width, 120mm in height, and 70mm in depth [7], yet the pin sets span from about 3mm in width, 11mm in height and 18mm in depth. When the entire lock is showing the users are able to find the key entryway with ease, but when the housing and cylinder were turned off (i.e., when the users want to feel just the pins and avoid the main constraints of the lock) the users sometimes had a difficult time locating the pins with the tip of the pick. Users would position the pick anywhere in space surrounding the pins even though they seemed to believe they were positioning the pick directly where the pins were located. Part of this may be due to the users not having a sense for the relative size of the pick and pins, and part of this may have to do with the camera view as depicted on the monitor not being aligned with the Omni device, (especially when the user changed the camera view). The authors anticipated this problem and so implemented the option, (already mentioned in section 3.1.1), to constrain the user and pick to the plane of the pins. When the 'm' key on the keyboard is pressed the user can toggle between turning this feature on and off, and thus can choose to be pulled to the middle plane perfectly aligned with the pins. With this feature enabled it becomes very easy for users to locate the pins with the tip of the pick.

Once a user has found a pin and wants to interact with it, (by pushing up on it from below), the reactions in the simulation are fairly realistic. When the user moves a pin vertically the pin gets pushed back down by the spring force above the pin. But, because of a slight delay in the interaction computations, the smallness of the pin masses, and the discrete nature of using a computer for a simulation, the pin tends to jump up and down on the pick tip as it tries to reach equilibrium between the spring above and the pick beneath. Fortunately, this effect disappears when the user is actually trying to pick the lock, because the tension wrench provides sufficient damping to resist the pin's motion. (Recall that the harder one pushes on the tension wrench the more difficult it is to move the pin.) When the pick tip is pulled away from the pin, and the wrench is also freed, all the pins will drop to the base of their workspace. The pins are kept within their workspace by resetting their position, velocity, and acceleration once they reach, or extend below, their initial position.

3.2.3 Creating and Importing 3D Models

To make the simulation appear more realistic the models were based on an actual lock set that had been disassembled. Each part shown in figure 6 was made to scale using a computer aided modelling software package called SolidWorks [2]. The models were then exported to sterolithography, (or *.STL), file format commonly used for Computer Numeric Controlled (CNC) manufacturing equipment (i.e., the equipment that the authors have access to at the University of Utah). Using a mesh converter [8] the STL file was then transformed into an object, (or *.obj), file format that could be rendered using Chai3D. (Note that Chai3d also supports object files of 3D Studio, or *.3ds, format.) A conversion scale was calculated for each object so that they would be rendered at actual size within the program. Note that the springs were not modelled, but that a virtual spring force was calculated and applied based on knowledge of linear spring properties.

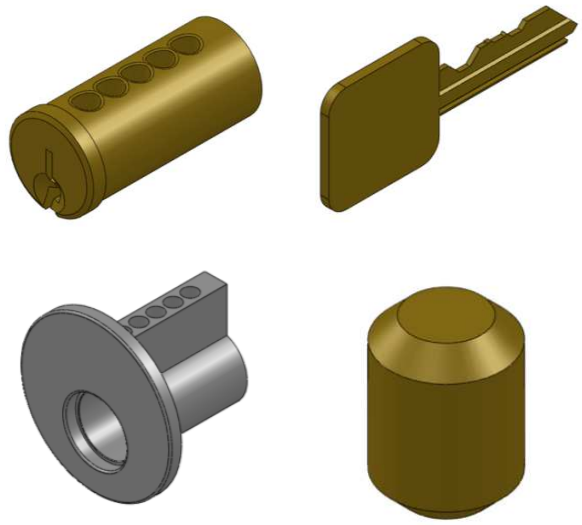

Figure 6: Various SolidWorks models used simulation: cylinder (top left), proper key (top right), housing (bottom left), offset pin (bottom right).

3.2.4 Using Chai3D to Implement the Simulation

Chai3d uses OpenGL (i.e., open-source graphics library) to visually render environments. By adding objects to a scene graph or 'world' and assigning meshes to each object, and positioning, orienting, and scaling each object, the entire scene graph can be rendered with a single command. Chai3d provides threads that allow different code loops to be run simultaneously, and at different rates and priorities. For instance, the graphics loop only needs to run at about sixty Hertz (i.e., cycles per second) to match human vision, but the haptics loop should run at about a thousand Hertz since this is greater frequency than adults can sense vibrations [5], and because objects are prevented from penetrating too deep within one another if the time steps are small.

Chai3d provides classes to encapsulate devices (such as the PHANToM Omni and the Novint Falcon); three-dimensional vectors and three-by-three matrices with the corresponding mathematical algorithms for quick vector and orientation computations; and many other useful objects, algorithms, and features for rendering graphic and haptic environments. Chai3d offers both axis-aligned bounding boxes (AABB) and object oriented bounding boxes (OOBB) for collision detection, with the option of assigning one or the other to each object. These collision detection algorithms minimize the computations needed to quickly discover colliding objects before directly comparing object mesh triangles for the specific collisions. For the haptic devices it provides both a proxy and actual tool position in space. The proxy is a small sphere placed at the tip of the tool which is constrained to stay outside all objects, and when the haptic interface point (or HIP) penetrates an object the proxy stays at the object's surface, giving the impression that the object is stiffer than it may actually feel [6]. The force applied to the haptic device is then computed by taking the difference between the HIP and proxy positions in space, placing a virtual spring between the two points, and computing the force needed to pull the two points together. Frictional effects are also taken care of by Chai3d. This command is as simple as assigning static and dynamic friction coefficients.

Among Chai3d's example files are several demonstrations that show dynamic interactions between objects in various virtual environments. But, the authors were unable to get a working dynamic environment for the imported objects (perhaps because the masses were too small at the time of the attempt, or perhaps because Chai3d seems to base its calculations on the first three detected contact points between each object). Instead of using the physics engine provided by Chai3d, the authors fixed the housing and cylinder in space then used Chai3d's collision detection to find collisions between the pick tip and the offset pins.

The normal force, N, applied by the haptic paddle against the pins was used to calculate the friction of the pins (i.e., $F = N \cdot \mu$), and the spring force was calculated based on the displacement, δ, of the pins within the shaft and the spring constant, K, as measured using the actual model (i.e., $F = K \cdot \delta$). If the user was applying a force to a pin, then this was calculated based on the proxy and HIP positions. Thus, the total force acting on each pin was calculated for each haptic loop, thus ensuring smooth flow of the pin dynamics. The trapezoidal Rule was used to numerically integrate the accelerations, velocities, and positions of each pin. If the pins went out of bounds, either above or below, then the current and previous positions, velocities, and accelerations were reset.

3.2.5 Programming Pin Motion

Because of the slight time delay in haptics of the system (which becomes apparent through the haptic paddle as soon as the pick tip comes into contact with any object in the environment), together with small-scale masses (ranging from 0.26mg to 0.43mg as calculated by SolidWorks), imprecise time measurement, and relatively large forces (with respect to the pin masses), the pins originally oscillated at high frequencies. In order to slow these oscillations the pin masses were increased by three orders of magnitude (since, according to Newton's Second Law, mass resists those accelerations caused by applied forces). With this change the pins behaved more like those within an actual lock, but felt heavier when being lifted. Despite this discrepancy between the real and virtual environments, the experience of virtually picking a lock remains fairly realistic. When the tension wrench is applied the pins do not oscillate, and the experience stays true to reality.

Because rendering the lock environment in Chai3d can be accomplished with a single command, the rotations and spatial positions cannot be directly applied as when using OpenGL directly. Instead, in order to rotate the cylinder and offset pins, (as each pin is compromised), it becomes necessary to also change the position of the origin of each of these objects. Currently this feature is not implemented, but is high on the priority list for new features.

3.2.6 Graphic and Haptic Computational Separation

For this simulation only two threads were used, one for the haptics and one for the graphics. The haptic paddle was created as a separate environment in which only its haptic method was called. Thus, when the main haptic thread was running it would request the haptic paddle to update its forces (calculated using a proxy, virtual wall, and spring constant), and the scene graph to update its object positions and orientations. With this information, the collisions of the pick tip with the environment were computed to find the user's applied force. With the user's force thus computed, the pin dynamics were then updated, and the timer was set to wait until the next expected cycle so that only a thousand cycles occurred within each second of time.

Although the haptics was being displayed at a thousand Hertz, the graphics only needed to be displayed at about sixty Hertz. In order to get the object positions and sensations to appear the same only the haptics thread was allowed to alter data such as global variables, and the graphics thread was only given power to extract this data to render it graphically within its own thread.

Button listeners were also added to the simulation so as to detect when the keyboard buttons were pressed. This enables the user to toggle certain features of the environment as mentioned in section 3.2.

3.2.7 Coupling Haptic Devices Together

In order to better mimic the experience of picking a lock it was desirable for the simulation to recognize when the pin separations were aligned with the housing-cylinder shear-line and to trigger a breakpoint as each pin was compromised. Within the haptic paddle environment is the option to communicate when a pin has been compromised. Thus the two haptic devices would be able to feel when each pin is compromised. Unfortunately, this coupling did not get implemented in this version, but is high on the priority list of simulation improvements.

4 FUTURE WORK

Currently the pin tumbler lock-picking simulation accurately mimics certain features of the actual lock-picking experience, but it is not complete. This work will be continued in order to improve the simulation and prepare for haptic demonstrations at the University of Utah and conferences. Listed below are some of the main issues that remain incomplete.

4.1.1 Compromising a Pin

Currently, users are allowed to move the pins anywhere within the pin workspace, and can align the pin separation with the shear-line. The user can also feel the sensations that occur through the wrench when a pin is compromised. But, currently these two features are not coupled together. Instead of relying on the keyboard option 'c' to stimulate a pin compromise, the program needs to identify when the pin separation and shear-line match within a certain tolerance, and communicate to the haptic paddle when this condition has been reached for each pin.

4.1.2 Multiple Collision Points

The simulation only uses the pick tip to interact with the virtual environment. This can destroy the sense of immersion when the user sees the rest of the pick moving through the lock assembly either vertically, or at an angle. In order to make the simulation more realistic it is desirable to attach multiple collision points to the pick, rather than at a single point. The proxies would keep the pick oriented properly within the enclosed space, and would add to the reality of this picking interaction. This feature would also allow the pick to manipulate multiple pins at the same time as when pressing the pins up using the pick shaft.

4.1.3 Multiple Picks and Working Key

With multiple collision points enabled on a given pick it would be easy to add several types of picks to work with when picking the lock in this environment. One could gain practice using various picks and methods, or even test pick designs. Another option that could be easily implemented with multiple collision points would be a working key as one kind of pick. Then, when inserted, this

key would enable the cylinder to be turned immediately. Multiple improper keys could be added to demonstrate that only one key would open the lock.

4.1.4 Pin Independence and Complex Interactions

Obviously, if the pins should be able to separate at the shear-line, then they must be separate objects. Indeed, in creating this simulation, the models for the driver and offset pins were modeled as separate objects, yet were combined for the simulation as though they were glued together (as can be seen in figure 5). For the simulation to be more accurate, each pin, (all driver and offset pins), should interact with each other and the cylinder individually. In that case it would be allowable to let the driver pin stay placed, but pull the offset pin down using gravitational forces. It must also be allowable that the offset pin go too high in the shaft and get stuck under the frictional force. Also, when the offset pin is pushed against the driver pin, this should be the only time that the driver pin moves (in a chain of applied and reactive forces). It would also be nice to add mushroom pins to the system. While these features may sound simple, their implementation will take some work, and the logical checks will further delay the haptics of the system.

4.1.5 Camera View Singularity

When moving the camera view around the lock assembly there are two viewpoints that cause the visual display to jump suddenly by pi radians because of singularities present in the computation of the viewing matrix. These two points occur directly above and below the lock. Currently these two viewpoints are considered rare enough that they could be ignored. But, to make a more complete simulation it would be better to remove these singularities.

4.1.6 Cylinder and Wrench Rotations

As each pin is compromised the program causes the cylinder to rotate slightly. These rotations were intended to be about the cylinder's axis of rotation, but instead ended up rotating about the objects' origin (i.e., the location in space corresponding to where x-, y-, and z-coordinates of the *.obj file are simultaneously zero). Also, the wrench should rotate about the same axis of rotation at the same time as the cylinder rotates. These rotations should be implemented with the pin compromises to maintain a sense of immersion with this virtual environment.

4.1.7 Textures

Currently all objects appear white in the lock picking simulation, but in reality they should appear as either shiny or dull brass (for the housing, cylinder, and pins), or nickel (for the wrench, pick, and key). Also, it would be additionally nice, though not terribly useful, to have reflections on the shiny part of the housing faceplate.

4.1.8 Randomized Pin Compromises

In an actual lock not all pins can be compromised in any order. If the pins are offset slightly, then one pin will be pressed against the cylinder first, and only it can be compromised at that time. Additionally, the foremost pin (i.e., the pin visible from the outside of the lock) is frequently the last one that can be picked. In order to further match reality, and to provide an added challenge that is normally present in picking a lock, it would be desirable to implement a randomization as to the order the pins must be compromised.

4.1.9 Generalized Implementation

While it is nice to be able to see within a pin tumbler lock while picking it, for this simulation to be truly marketable for locksmiths and lock designers this simulation program would need to be generalized to accept any kind of lock, and automatically scale and place the lock parts within the environment.

4.1.10 Experimentation

In order to evaluate how successfully this simulation compares to actually picking a lock, it would be desirable to perform a series of experiments using both amateur and expert lock-pickers, and obtain user feedback about how the simulation performs and how it could be improved.

5 CONCLUSION

The goal of creating a haptic pin tumbler lock-picking simulation has been accomplished using a haptic paddle, PHANToM Omni, a visual display, and libraries from Chai3d and OpenGL. By providing threads to separate the haptics from the graphics, computational resources needed to implement this simulation were minimized. In the simulation the user can feel the force feedback they would experience when actually picking a lock, as well as the sensations that occur when each pin is compromised. Mostly realistic pin dynamic behaviour has been implemented within the simulation by increasing the mass of the pins. The pins respond to ever-present spring forces above, the frictional forces provided through the haptic paddle, and the interactions that the user provides from the PHANToM Omni. True to reality the pins are more difficult to move when the force applied to the haptic paddle (or tension wrench) increases. Thus, the two haptic displays provide realistic force feedback.

While additional work is needed to make the simulation more realistic, complete, and capable of dynamically interacting with any lock assembly, users who have tried this simulation seem to enjoy the experience, and are not bothered by the discrepancies between the virtual with real experiences.

REFERENCES

[1] H. Conkel. *How to Open Locks With Improvised Tools.* Level Four Publications. 2001.

[2] Dassault Systemes SolidWorks Corp. SolidWorks Educational Edition 2010-2011.

[3] S. Hampton. *Advanced Lock Picking Secrets.* Paladin Press. 1989.

[4] S. Hampton. *Secrets of Lock Picking.* Paladin Press. 1987.

[5] R. L. Klatzky and S. J. Lederman. (Prepublication version). Touch. In A. F. Healy and R. W. Proctor (Eds.), *Experimental Psychology*. (pp. 147-176). Volume 4 in I. B. Weiner (Editor-in-chief) *Handbook of Psychology*. New York: Wiley.

[6] Provancher. Notes from haptics class, Lecture 4: Haptic Rendering Basics, slide 18. 2011

[7] Sensable. Phantom omni technical specification. Phantom Omni Device. Retrieved 7 May 2001 from http://www.sensable.com/haptic-phantom-omni.htm

[8] SYCODE. Mesh Converter 1.0. 2008.

[9] L. N. Verner and A. M. Okamura. Sensor/actuator asymmetries in telemanipulators: Implications of partial force feedback. In *Proc. of 14th Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems*, pages 309-314. 25-26 March 2006.

[10] L. N. Verner and A. M. Okamura. Telemanipulators with sensor/actuator asymmetries fail the robustness criterion. In *Proc. of 16th Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems*, pages 267-271, March 2008.

[11] Precision Graphics. Modified version of image retrieved 5 May 2011 from http://www.yourdictionary.com/lock