Toward Modeling of a Suturing Task

Matt LeDuc Shahram Payandeh John Dill

Experimental Robotics and Graphics Laboratory
School of Engineering Science
Simon Fraser University

Abstract

In this paper we present our initial work on simulating suturing using mass-spring models. Various models for simulating a suture were studied, and a simple linear mass-spring model was determined to give good performance. A novel model for pulling a suture through a deformable tissue model is presented. By connecting two separate tissues together by way of the suture, our model can simulate a suturing task. The results are shown using software we developed that runs on a standard PC and models the action of two suturing devices commonly used in minimally invasive Laparoscopic surgery.

1 Introduction

In this paper we attempt to model a suture, and create a simulation of a suturing task realistic enough to use in a surgical training environment, and fast enough to run on a desktop computer. One of our main goals is for the system to run on PC hardware, i.e. a Pentium III 933MHz system with an NVIDIA GeForce 2 video card. Such a goal is difficult to achieve since simulating deformable objects and performing collision detection are both computationally intensive. However, for development of a surgical training environment, e.g. our Laparoscopic Training Environment (LTE), virtual representations need not be extremely precise. They only have to be accurate enough to facilitate a trainee in gaining the required dexterity and hand-eye coordination. In this paper we model a suture and a simple deformable object representing tissue. We also develop and describe two tasks which use the suture and tissue models as well as two simulated devices that the operator uses to stitch together pieces of tissue.

Many groups have been working on surgical simulation in general ([1, 2, 3, 4]), as well as the specific task of simulating suturing ([5, 6, 7, 8, 9]). Measuring surgeons' performance using a simulation has been investigated [5] by focusing only on the initial penetration of the object by the suturing needle, and not considering the entire suturing process. A group at Rice University took the opposite approach [6] and focused only the realistic simulation of a suture and its behavior, while not looking at the actual suturing task. Their paper describes a method for simulating a suture using a spline of linear springs and large overlapping nodes. Although their method gives up some speed and stability, they are able to tie various types of knots in the suture material.

Suturing itself has been explored by many groups. Webster et al. created a simulation that is based on a 2D mass-spring model [7]. Unlike ours though, their tissue model appears to be restricted to a 2D plane, with the feedback forces being calculated based only upon the depth and angle of the needle as it penetrates that 2D plane. Brown et al. designed a system [8] for training surgeons in the task of suturing blood vessels. The suture was simulated using rigid links of a fixed length while the blood vessels themselves were simulated using mass-spring systems. The same group also designed a software framework that supports many different kinds of surgical tasks [9]. Unrelated to surgery simulation, but using similar mass-spring technology, various legless animals have been simulated [10]. This method could possibly be used in surgical simulations to create realistically moving organs, such as the heart and lungs.

This paper presents an initial novel approach for simulating a suturing task, where the suture and needle are passed between two tissues in order to connect them. The paper is organized as follows. Sections 2 and 3 describe the deformable models we used to represent the objects in our simulation (the suture and tissue models), while section 4 describes the algorithm used to simulate the suturing. Sections 5 and 6 outline two demonstrations we developed using the techniques described in the previous sections, while Section 7 discusses possible directions for our future research.

2 Deformable Objects

Triangular surface meshes represent both the rigid and deformable objects in our virtual environment, i.e. the suture, the tissue, and the Endo Stitch and Needle Driver devices, both of which are used to perform suturing in laparoscopic surgery (see sections 5 and 6).

Our deformable models are mass-spring models. Mass-spring models, along with finite-element models, are well known ways of simulating deformable objects [6, 7, 8], so we limit our discussion to those aspects especially relevant to our development here.

Each node in our triangular mesh has a mass associated with it, and each triangle edge has a coincident spring, an "*edge spring*", joining together the two nodes that define the edge. When stretched or compressed, each spring in our models apply a force to the attached nodes of

$$F = K_e(L - L_0)\frac{(P_a - P)}{|P_a - P|}$$

where L and L_0 are the current and rest lengths of the spring respectively, P is the position of the node who's applied force is being calculated, P_a is the position of the other node of the spring, and K_e is the elasticity of the spring.

2.1 Home Springs

Using only a mass-spring surface model, one could not construct 3D deformable objects that could be compressed and stretched, since they would not return to their initial shapes after deformation. One approach to solving this problem is to create an internal structure using a set of springs to give the surface the support needed to maintain, and return to its initial shape after being deformed. For example, this method has been used to model blood vessels [7]. Although it proves effective and stable for small models with small displacements, with more complicated objects or large deformations, the object can easily become unstable or permanently tangled.

To address the problem of maintaining an object's shape, our models use "*home springs*" connected to each node. These zero rest length springs connect each node to a fixed location in 3D space and maintain the connected vertex in its undisplaced position through the creation of an internal force proportional, but of opposite direction, to the displacement of the node from its "home" location. As a result, when the deformable object has been deformed, for example by an interaction with another object, after the interacting object has been removed, the deformed object will be restored back to its original shape (for example, the square in Figures 1a and 1b).

We have used this method before in an early phase of our LTE as well as in a surface mesh subdivision model in [11]. It is an efficient solution since the force applied by each home spring to its connected node is simply calculated as $F = K_h(H - P)$, where H is the home position of the node. This equation consists of only a vector subtraction, and scalar multiplication, and is therefore much faster than the one used for the edge springs, which involves a square-root operation plus a divide. Since the number of home springs in a surface model will be proportional to the number of edge springs, this model adds only a small constant amount of computation to the basic the mass-spring surface model.

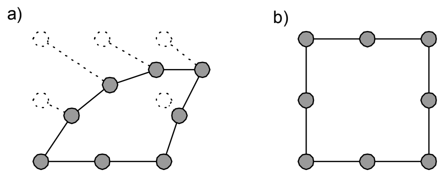


Figure 1: Deformation and restoration of a model containing home springs

2.2 Node-position Integration Method

To solve for the deformed state of the object we use Euler's method to integrate the positions of the nodes according the to the following equations, where M is the node's mass, B is a damping constant, dt is the timestep, and F_i, V_i, and P_i represent the force acting on the node, the node's velocity, and the node's position, all at time step i.

$$F_i = \sum(\textit{spring forces}) - BV_{i-1} \qquad eq.\ 1$$

$$V_i = V_{i-1} + \frac{F_i}{M}dt \qquad eq.\ 2$$

$$P_i = P_{i-1} + V_i dt$$

Combining equations 1 and 2, we get

$$V_i = V_{i-1} + \frac{\sum(\textit{spring forces}) - BV_{i-1}}{M}dt$$

If we assume that the forces on the nodes are large compared to the node's kinetic energy, which consistent with tissue and the forces on it during suturing, then this equation can be simplified to:

$$V_i = \frac{\sum(\textit{spring forces})}{B}$$

In this method, the velocity of a node at a given point in time is calculated only from the forces acting on the node at that instant, and does not include the velocity at the previous time step (i-1). The advantages in using this quasi-static method are speed and simplicity. Since there are fewer calculations, it runs faster, 8-10% in our application, and also allows the mass attribute M to be left out of calculation.

3 The Suture Model

The suture uses the same deformable model data structure used for the deformable objects. The difference is that instead of creating a 2D mesh in 3D space, the nodes are simply arranged linearly, one after another, and joined together with edge-springs (see Figure 2). The result is a 1D suture in 3D space.

Because the suture must be able to move within the scene, the home spring constant K_h of its nodes is set to zero. We also want the suture to behave realistically under the influence of gravity, so a constant gravitational force is applied to each node.

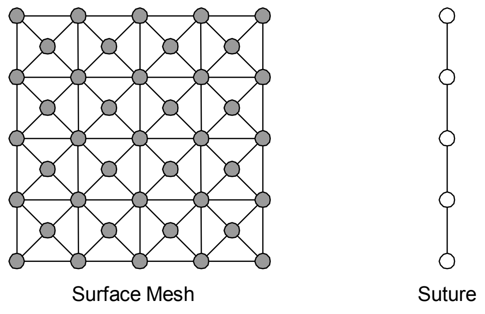


Figure 2: Surface mesh and Suture models.

We investigated several other possible representations of the suture, involving various forms of springs and dampers. The first, and simplest one, was simply masses connected together by springs and involved no damping. The second model added dampers running between the masses. Three more complex, and more realistically behaving, models involving torsion spring, torsion dampers, and viscous damping effects were also implemented. We chose to use the first model for the suture in this simulation since it is less computationally intensive. It originally looked quite unrealistic due to the lack of damping in the model, but by using our quasi-static method for integrating the position of the model's nodes, a viscous damping is introduced without adding to the complexity of the calculations and slowing the simulation down.

3.1 Rendering the Suture

Since the nodes of the suture lie in a linear chain, an obvious rendering method is to simply render the suture as a series line segments. This is fast and simple, but would not be the same rendering method used by the triangle-based objects, and the two would therefore look very different.

To avoid this problem, we chose to render the suture by creating a flexible tube made up of triangles and containing the same number of sections as there are segments in the suture. We then reposition this tube over the suture before each frame is rendered. This newly defined shell is rendered instead of the suture itself. An illustration of the process can be seen in Figure 3. Since the suture is now rendered using a triangle model, it can undergo the same lighting calculations, and have an appearance consistent with the rest of the objects in the scene.

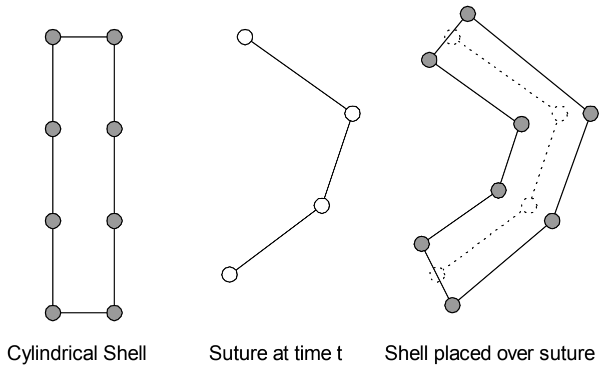


Figure 3: Suture rendering

4 Simulated Suturing

4.1 Basic Suturing Algorithm

In real suturing, as the needle passes through the tissue, it creates a hole through which the thread is pulled. As long as the forces pulling on the suture are small, friction between the suture and tissue will tend to prevent the suture from sliding through the hole and the suture will pull the tissue along with it as it moves. Simulating suturing by creating a small hole in the triangularly modeled deformable object, and then simulating the friction forces between it and the suture would be overly complex.

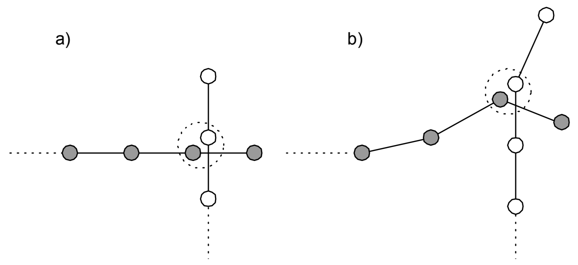


Figure 4: Simulation of the suture running through a small hole in the object.

As an alternative to the above mentioned complex method, we model the effect of a suture passing through a hole by treating one of the nodes of the tissue model as a hole, and connecting this node to one of the nodes of the suture. This can be seen in Figure 4a, where the filled circles are the nodes of the tissue, and the hollow circles are nodes of the suture. In Figure 4a, there is no force being applied to the suture. In Figure 4b, a force is applied. This force pulls the suture toward the upper right. Since the node of the suture is joined to a node of the tissue, the two move together as one, and the rest of the tissue gets pulled along with it.

When real tissue gets pulled along due to the friction between it and the suture, there is a limit to how far it will move. Eventually the forces within the tissue will become large enough to counter the friction force and cause the tissue to slide down the suture.

In Figure 5a, node N of the tissue model (the hole through which the suture has been pulled) is being pulled down by its neighboring nodes; however, the friction forces from the suture balance the downward force. Once the suture and tissue have been stretched enough, the required force from the suture to the tissue in order to keep it from sliding will be greater than the friction between them. To simulate this sliding, node N is detached from node S_0 of the suture, and reattached to node S_1. If the suture continues to be pulled, then node N will continue to slide down the thread (Figure 5b), creating the impression the suture is slipping through a hole.

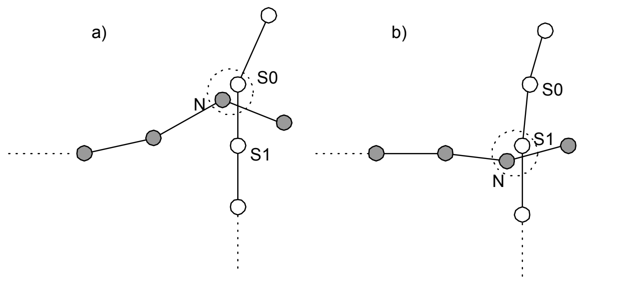


Figure 5: Slipping of the deformable object down the suture.

4.2 Multiple Slipping

During suturing, pieces of tissue will be pulled together by a suture. In Figure 6a a suture is shown between two pieces of modeled tissue. In order to stitch the two pieces together, the suture will first pass through the left object and then the right. Using the suturing algorithm of section 4.1 can lead to the situation shown in Figure 6b where two tissue nodes will be attached to the same suture node. Since the current algorithm has no inter-object, or self-collision detection between the deformable models; therefore, a method is needed to ensure that the two tissue nodes on the suture are not able to slide past each other. For example, in Figure 6c the right piece of tissue will be under more strain than the one on the left, and it will be the first to slide. Because it lies above the left piece of tissue on the suture, it can not slide without also pulling the left piece with it.

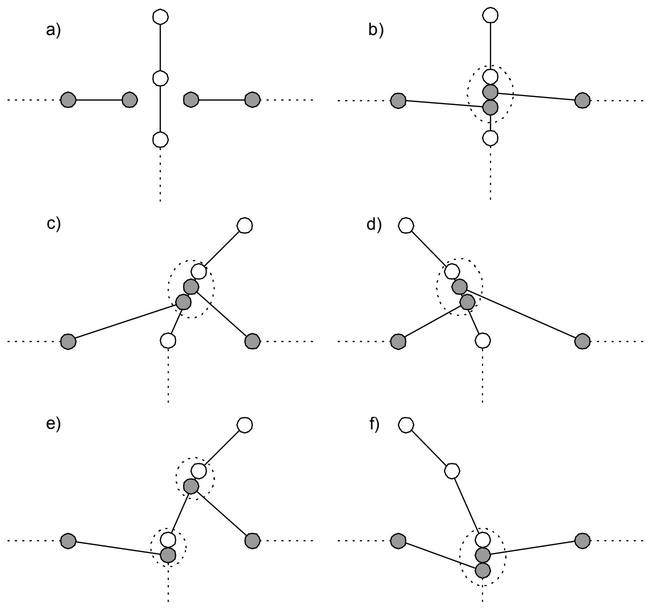


Figure 6: Multiple objects sliding down the suture.

To handle this situation, for each suture node we store an ordered list of tissue nodes that are attached to it. This linked-list approach allows us to maintain the order in which the tissue nodes were pierced by the suture. This information allows us to handle situations such as those shown in Figures 6c and 6d. In Figure 6c the left piece of tissue has been stretched further that the one on the right, is therefore under more strain, and if under enough tension will slip down the suture leaving the other tissue node behind (Figure 6e). In Figure 6d the right piece of tissue is under more strain; however, it cannot slip without pulling the left tissue's node with it (Figure 6f). This can happen only when the force on the right tissue's attached node is large enough to overcome the friction between itself and the suture, and the combined force of the attached nodes is enough to overcome the combined friction between the nodes and the suture. If this is not the case, then the tissue nodes will not slide.

It must be noted that even though several tissue nodes can be attached to a single suture node, the opposite is not true. Attaching a single suture node to a single tissue node can lead to situations involving two tissue nodes attached to the same two suture nodes. Trying to determine when the tissue should and should not slip, and whether it should take the other suture node with it is very difficult to solve. We have chosen simply to not allow this, and the method used for intersecting the needle with the tissue (section 4.3) reflects this decision.

4.3 Attaching a Needle

The needle is modeled as a rigid polygonal model which either moves with the device gripping it, i.e. the Endo Stitch device or needle driver, or moves freely under the forces applied to it by the segment of suture attached to its non-pointed end and by the tissue the needle may be penetrating. If the needle is in the grip of the needle driver device and a force feedback device is being used, these forces can be used to provide the user with force-feedback.

In our model, the suture is not simply attached to the end of the needle as would normally be seen in suturing and sewing, Instead, the first few nodes of the suture are forced to lie along the center of the needle's current position in the scene (see Figure 7). The needle is then rendered over top of the suture. This method avoids the need for a special algorithm to allow the tissue being stitched to slide along the needle (or equivalently the needle through the tissue); the same algorithm that lets the tissue slide along the thread (sections 4.1 and 4.2) can be utilized.

The collision detection between the needle and the tissue being sutured is only calculated at the tip of the needle, i.e. the first node of the suture. When the tip of the needle passes through a triangle of the tissue model, the nearest of the triangle's vertices that isn't already connected to one of the vertices of the suture is then attached to the tip of the needle (the first vertex of the suture). If all of the triangle's vertices are already attached to the suture then the triangle is subdivided, resulting in a new vertex being created. That vertex won't be attached to any suture node and can therefore be attached to the tip of the needle.

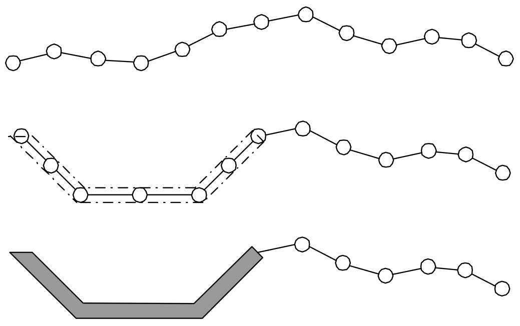
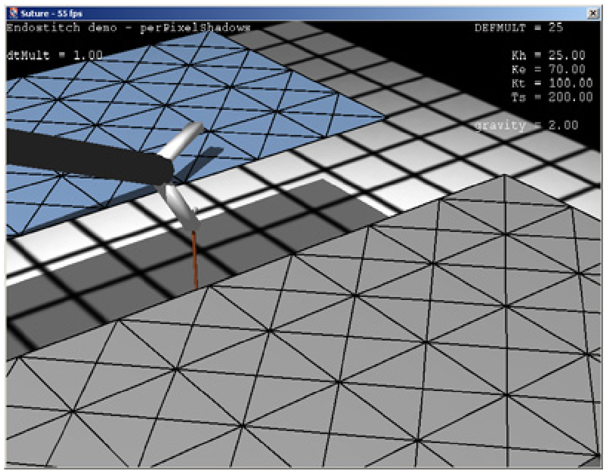


Figure 7:Attaching a needle to the suture.

5 Endo Stitch Suturing Task

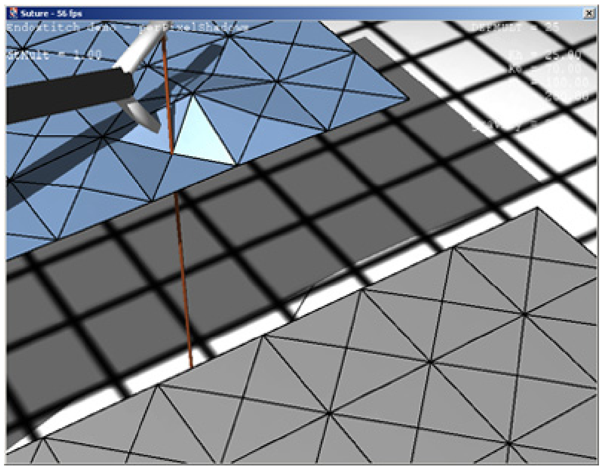
For the initial test of our suturing algorithm, we modeled an Endo Stitch device and used it to perform suturing. To simplify matters, collision detection and force feedback were omitted. Also, since the needle is always attached to this type of device, it didn't need to be able to move independently due to forces from gravity, the suture, or the tissue the needle may be penetrating. This further simplified the implementation.



```
Suture - 55 fps
Endostitch demo - perPixelShadows          DEFMULT = 25
dtMult = 1.00
                                              Kh = 25.00
                                              Ke = 70.00
                                              Kt = 100.00
                                              Ts = 200.00
                                         gravity = 2.00
```

Figure 8: Jaw with needle is under the tissue.

The end of the Endo Stitch device has two jaws, and through the activation of a mechanical switch, can pass a needle between them. With the needle on one of the jaws, the surgeon can pierce the tissue. Closing the jaws and activating the switch will pass the needle passed to the second jaw, pulling the suture through the puncture. We simplified the operation of the virtual Endo Stitch device slightly: a single key press on the computer's keyboard will close the jaws, pass the needle across, and then open the jaws again having passes the needle through any tissue in the way.



```
Suture - 56 fps
Endostitch demo - perPixelShadows          DEFMULT = 25
dtMult = 1.00
```

Figure 9: Needle has passed from one jaw to the other, pulling suture through first object.

In this task we defined two simple deformable objects to represent pieces of tissue. To suture the two pieces of tissue together, one positions the device so that the two jaws of the device are on opposite sides of one of the pieces (Figure 8). Pressing the keyboard key will pass the needle through the tissue to the other jaw. Raising the device pulls the suture through (Figure 9 Performing this same process on the other piece of tissue, and pulling on the suture slightly, will bring the two pieces of tissue together (Figure 10). To continue stitching, the process is simply repeated.

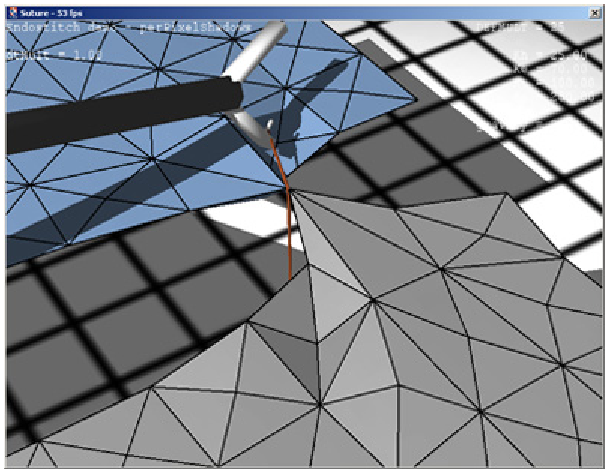


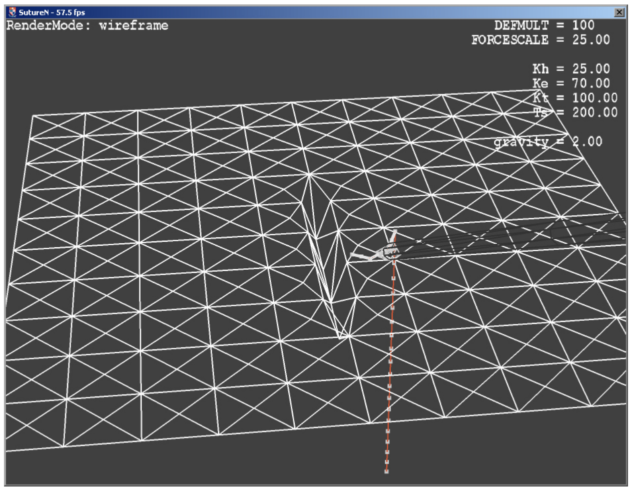
Figure 10: Objects pulled together after passing suture through second object.

6 Needle Driver Suturing Task

The second training task is based on open surgery, as opposed to laparoscopic surgery. This task contains the features not implemented in the Endo Stitch training task of section 5. When the grippers open the needle is released and will either fall under the effects of gravity and the attached suture material, or if the needle is embedded in the tissue, it will remain there. If the grippers are then closed and passed through the needle, the latter will be picked up again. Collision detection between the needle tip and tissue has been implemented and the needle can be pulled back out of the tissue.

The basic surface mass-spring model serves as the patient model, with a depression added to model the incision. The underlying structure can be seen clearly in the wire frame view of the simulation shown in figure 11. The suture can also, as expected, be seen passing through the needle. The model is texture mapped to add detail to the incision and skin surfaces (figure 12).

The suturing in this task is performed in a manner similar to that of the Endo Stitch task. The needle is first passed through the tissue on one side of the incision and then the other. To pass the needle through the tissue, the needle is inserted in such a way that the needle's tip exits the tissue. The user must then let go of the needle and grasp it again buy its exposed tip. Continuing to pass the needle through from side to side, and pulling the suture tight, will close the incision (figures 13 and 14).

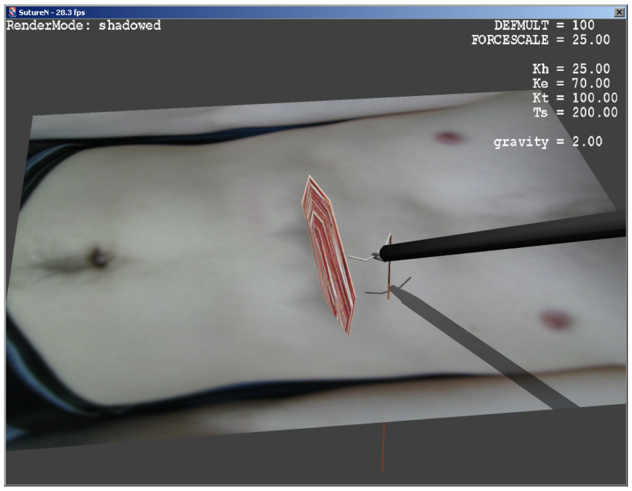


```
RenderMode: wireframe                    DEFMULT = 100
                                    FORCESCALE = 25.00

                                          Kh = 25.00
                                          Ke = 70.00
                                          Kt = 100.00
                                          Ts = 200.00

                                     gravity = 2.00
```

Figure 11: A wire frame view of the Needle Driver task.

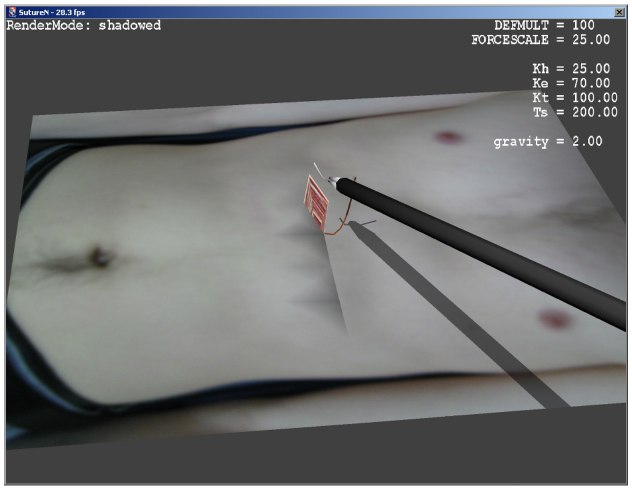


```
RenderMode: shadowed                     DEFMULT = 100
                                    FORCESCALE = 25.00

                                          Kh = 25.00
                                          Ke = 70.00
                                          Kt = 100.00
                                          Ts = 200.00

                                     gravity = 2.00
```

Figure 12: An incision across a patient's chest.

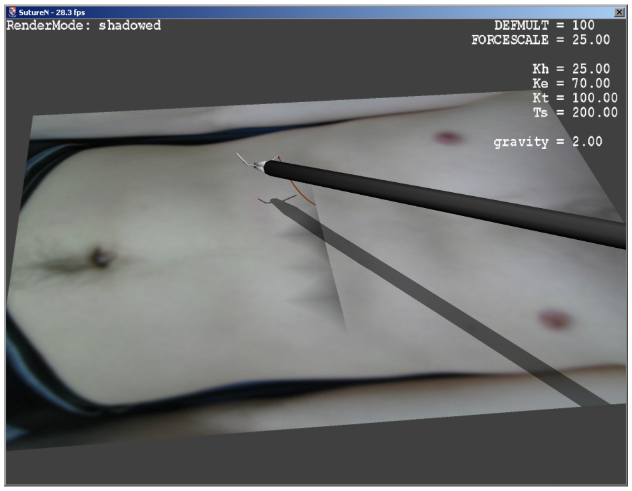


```
RenderMode: shadowed                     DEFMULT = 100
                                    FORCESCALE = 25.00

                                          Kh = 25.00
                                          Ke = 70.00
                                          Kt = 100.00
                                          Ts = 200.00

                                     gravity = 2.00
```

Figure 13: The incision after several stitches.


SutureN - 28.3 fps

```
RenderMode: shadowed
                                   DEFMULT = 100
                               FORCESCALE = 25.00

                                     Kh = 25.00
                                     Ke = 70.00
                                     Kt = 100.00
                                     Ts = 200.00

                                 gravity = 2.00
```

Figure 14: The closed incision.

7 Future Work

One particular area we plan to pursue is improvements to the collision detection used in this simulation. Ideally we would like to have the tool be able to touch and deform the objects being sutured, and have the suture be able to rest on the surface of the objects instead of passing through them. Having self-collision detection, i.e. allowing objects to collide with themselves, could lead to the ability to tie knots in the thread, and/or perform more complex types of suturing.

Also, the current mass-spring model is too stretchy for use as the suture. This stretchiness is both unrealistic, since real suture material is fairly inelastic, and also makes it difficult for the user to pull enough of the suture through the tissue to create the next stitch. Using a higher spring constant to reduce the stretchiness is likely not a suitable solution since it would greatly increase the instability of the system. The tissue model could also be improved. For example, using a tetrahedral mesh, instead of a the current 2D triangular surface mesh, might result in more realistic results.

Lastly, a proper simulation of suturing and knot tying during laparoscopic surgery needs to be developed, as opposed to the open surgery task of section 6. It would also be useful to perform a user study on it to evaluate its potential for improving a subject's suturing skills.

References

[1] P. Gorman, J. Lieser, W. Murray, R. Haluck, and T. Krummel, "Evaluation of Skill Acquisition Using a Force-Feedback, Virtual Reality-based Surgical Trainer", *Proceedings of Medicine Meets Virtual Reality 1999*, IOS Press, 1999, pp. 121-123.

[2] J. Berkley, S. Weghorst, H. Gladstone, G. Raugi, D. Berg, and M. Ganter, "Fast Finite Element Modeling for Surgical Simulation", *Proceedings of Medicine Meets Virtual Reality 1999, ISO Press*, 1999, pp. 55-61

[3] H. Delingette, "Towards realistic soft tissue modeling in medical simulation", *proc. of the IEEE: Special Issue on Surgical SImulation*, April 1998, pp. 512-523

[4] U. Kuhnapfel, H. Cakmak, H. MaaB, "Endoscopic surgery training using virtual reality and deformable tissue simulation", *Computers & Graphics 24 (2000)*, pp. 671-682

[5] Robert V O'toole, Robert R Playter, Thomas M Krummer, William C Blank, Nancy H Conelius, Webb R Roberts, Whitney J Bell, Marc Raibert "Measuring and Developing Suturing technique with a Virtual Reality Surgical Simulator", *Journal of the American College of Surgeons*, July 1999, pp. 114-27

[6] Andrew Ladd, "Simulated Knot tying", *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, Washington DC

[7] Roger W. Webster PhD, Dean I Zimmermanm Betty J, Mohler, Michael G. Melkonian MD, Randy S. Haluck MD "A Prototype Haptic Suturing Simulator", Proceedings of Medicine Meets Virtual Reality 2001, IOS Press, 2001, pp. 567-569

[8] Joel Brown, Kevin Montgomery, Jean-Claude Latombe, and Michael Stephanides, "A Microsurgery Simulation System", *Medical Image Computing and Computer Aided Interventions*, The Netherlands, October 2001

[9] K. Montgomery, C, Bruyns, J. Brown, S. Sorkin, F Mazzela, G. Thonier, A. Tellier, B. Lerman, A. Menon, "Spring: A General Framework for Collaborative, Real-time Surgical Simulation", *Medicine Meets Virtual Reality*, IOS Press, Amsterdam, 2002

[10] Gavin S.P. Miller, "The Motion Dynamics of Snakes and Worms", *Computer Graphics*, Volume 22, November 4 1998, pp169-178

[11] Jian Zhang, Shahram Payandeh and John Dill, "Haptic Subdivision: an Approach to Defining Level-of-detail in Haptic Rendering", *10th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, IEEE Computer Society, pp. 201-208