MODELLING A MICROSURGICAL SUTURE IN UNITY

Justin Stevens[1] , Chad Hogg[1] , Evan Hanzelman[1] , Brian Smith[2] , Joseph Sassani[2],[3]
[1]Millersville University
[2]Simulation Systems
[3]Penn State School of Medicine
{jmsteve1, chad.hogg}@millersville.edu, evan.hanzelman@gmail.com, bsmith@simsys.us, jxs14@psu.edu

ABSTRACT

Virtual reality simulators can be an effective way to train and evaluate future surgeons, but require realistic modeling of physical objects such as bodily tissues, sutures, and instruments. We have found accurately modeling a suture to be especially difficult, and describe here several approaches that we have attempted within the Unity game engine.

1 Introduction

Virtual reality simulators provide a cost-effective and safe way for surgeons to practice and perfect their craft, and to have their skills evaluated. We are attempting to build such a simulator, initially for ophthalmologic microsurgery, in the Unity 3D engine. So far we have succeeded in creating custom peripherals similar to real surgical instruments and sensing their location and orientation in space. An off-the-shelf 3D video system allows the user to see virtual representations of these instruments in a virtual world, in which maintain the same position and orientation as the peripherals do in the physical world. The goal of our simulator is that users will be able to use these virtual forceps to grasp a simulated suture to stitch together simulated simulated body tissues and tie knots, with the knot-tying as a particular point of emphasis.

The most challenging aspect of this simulation has been modeling a suture that behaves similarly enough to real sutures that the virtual experience will be meaningfully similar to the physical experience. For now, our suture is composed of a series of nodes connected by joints. We have found ways to configure these nodes and joints that produce a marginally acceptable model, but are continuing to attempt to improve it. The paper discusses the various challenges in creating a realistic model and the parameter values that we have found to work best.

2 Background and Related Work

Microsurgical suturing, or microsuturing, is performed by ophthalmologists in procedures such as scleral laceration repair and keratoplasty. It is one of the most technically demanding surgical skills, involving the precise placement of microsutures in sub-millimeter sized anatomy using only hand-held instruments. Consequently, microsuturing requires years of training and practice to master. In some cases, new technology has replaced the need for microsuturing. For example, cataract surgery is now generally performed via a self-sealing sutureless wound. However, procedures involving trauma repair, for example, still depend upon microsuturing, and it remains a critical part of the ophthalmologist's toolkit. Moreover, the reduction in the use of suturing during cataract procedures has decreased resident experience with this vital skill.

Therefore, the trend away from microsuturing creates a risk for those patients who must have a procedure that requires it. In routine clinical practice, ophthalmologists and ophthalmology residents now have fewer opportunities to perform "live" microsuturing than they once did, and in the absence of this frequent practice, skills can become rusty. In some cases, a resident may not acquire enough microsuturing practice during his or her ophthalmology training to gain proficiency, and therefore that resident may not be capable of operating independently or safely.

Unfortunately, opportunities to practice microsuturing outside the operating theater are limited. An optimal training method should (1) have high fidelity (i.e. realism or face validity), so that the rehearsed skills translate to real-world performance, and (2) be convenient to deliver, so that it can be implemented with meaningful frequency. These two attributes are often juxtaposed, and microsurgery training models rarely provide the type and frequency of practice needed for trainees to gain, maintain, and document proficiency [1; 2].

Current methods of microsuturing instruction include practice on tissue substitutes such as surgical tubing or 3D printed models; practice on cadaveric tissue and dead animal parts; and practice on live, anesthetized animals [3; 4; 5]. The inexpensive substitutes usually lack the fidelity necessary to mimic real-world conditions, whereas the more expensive biological models are low throughput, require regulatory oversight and specialized facilities, and cannot be repeated with meaningful frequency. While there are some validated measures of microsurgical skill [6], overall competency is judged largely by subjective evaluation. None of

these training methods offers objective assessment of performance, which is needed both for focused didactic feedback and for standards-based competency benchmarking. Thus, they require the presence of a trained observer to document and to score the trainee's performance.

At this time of extreme sensitivity to cost and quality in healthcare, improved training methods are a simple yet often overlooked way to drive efficiency. For microsuturing in particular, good technique is expected to contribute to better outcomes and shorter operating times. Therefore, the challenge we seek to address is how high-fidelity microsuturing instruction can be delivered frequently, expediently, and inexpensively.

Virtual reality-based simulation has begun to transform the delivery of surgical training. The advantages are plentiful: simulation exercises can be made to be very realistic, yet they are easily deployed in an office or computer lab; every aspect of a simulation can be measured, analyzed, and documented to evidence learning progress; multiple scenarios can be constructed to provide practice on varying clinical presentations; and scenarios can be re-run over and over, at zero marginal cost per repetition, until proficiency is achieved. Currently, sophisticated surgical simulators exist for numerous specialties and techniques [7; 8; 9; 10; 11; 12], with the most mature applications in laparoscopy and robot-assisted surgery. Within ophthalmology, we know of two mature simulators, the Eyesi Surgical Simulator [13; 14] and the HelpMeSee Eye Surgery Simulator [15; 16], that have had a significant impact on preparing surgeons for cataract and vitreoretinal surgery. However, no simulator has been widely accepted for microsuturing.

Creating a simulator entirely from scratch allows the most flexibility and control, but would make the project very large and complex. We have chosen instead to use Unity, a modern 3D game engine that provides various tools and components for modeling phenomena in virtual 3D worlds. Several other research projects have also built surgical simulators of varying types on top of Unity [17; 18; 19; 20]. None of these projects model knot-tying in sutures.

3 Unity Components

Unity allows the programmer to create Game Objects, which represent things that exist in the virtual 3D world. By default, Game Objects contain only a Transform component, which describes the object's position, rotation, and scale within the world. Many other types of pre-built components can be added to Game Objects and configured as desired. For example, a Mesh component makes an object visible and controls its appearance.

A Collider component describes the volume of space filled by the Game Object, and be used to detect intersections with the volumes of other Game Objects that also have Colliders on them. A Rigid Body component connects a Game Object to the physics engine, causing it to respond appropriately to forces such as gravity and collisions with other objects that also have Rigid Bodies. Joint components connect Rigid Bodies together, causing forces applied to one to propagate to others, and constraining the types of rotations and movements that are possible between the Rigid Bodies. Unity currently offers Character, Configurable, Fixed, Hinge, and Spring joints. The Character, Fixed, Hinge, and Spring joints are all special cases of Configurable Joints since they can all be modeled using the Configurable Joint with certain parameters. Physics Material components further allow the programmer to control how Game Objects interact with the physics system by setting properties such as the Game Object's coefficients of static and dynamic friction.

Game Objects can be created, and components added or removed from them and reconfigured both through a world editor interface and through scripts that run dynamically as the simulation is in progress [21].

4 Modeling Requirements

Our simulation requires three types of objects: a tissue subtrate, two pairs of forceps, and a suture. Prospective surgeons practicing inside the simulation will be expected to use the forceps to grasp the suture, pull it through the tissue subtrate, and tie a knot with the two ends of the suture.

The only interesting behavior that the tissue substrate must have is allowing the end of the suture to pass through it and pull the rest of the suture along with it. We have a simple substrate model with holes that seems sufficient for our purposes, at least for an initial simulator.

The forceps are made up of many parts, each with Meshes and Colliders and Rigid Bodies that are intended to appear and behave like real surgical instruments. They are the virtual representations of custom peripherals we created that allow the positions and rotations of physical objects the user is holding to be observed and updated in the virtual world in real-time.

The most challenging object to model has been the suture, and we have not yet found a solution that is entirely satisfactory. A real suture used in microsurgery behaves similar to fishing line, though it is much smaller. It should be flexible enough to easily bend when force is applied to it, yet rigid enough that it attempts to return to an entirely relaxed position in the absence of any forces. When forceps tug on an end of a suture, the force should be applied evenly across the length of the suture, both deforming and moving it. When a suture has been tied into a knot, the force of friction must be sufficient to prevent the knot from unraveling.

5 Suture Models

We have chosen to model the suture as a long chain of discrete Game Objects, each with Capsule-shaped Colliders and appropriate Rigid Bodies and Physics Materials, with Configurable Joints connecting each node to the next. Configurable Joints were chosen to give the developers the most freedom when creating the joints, since they offer everything the other joints have and more. When force is applied to any node, that force is distributed through the joints to the other nodes. These nodes connected by joints create a chain-like structure shown in Figure 1.

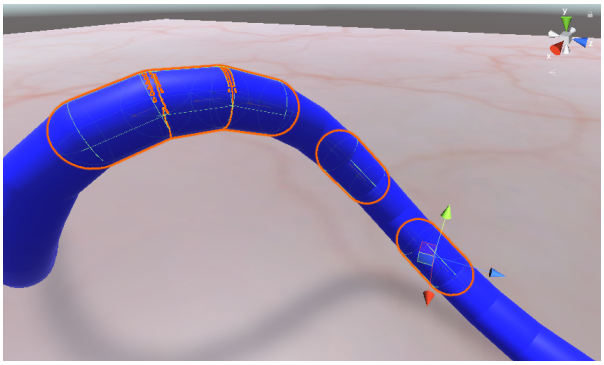

Figure 1: From left to right, three consecutive suture nodes, then two other nonconsecutive nodes.

Currently, this suture is being modeled using 128 individual nodes with 127 separate Configurable Joints connecting them. A sequence of more numerous but smaller nodes might allow better behavior, but increases the computational load on the physics engine, which must respond immediately to changes in the system. When these nodes are given the ideal parameters, which we have yet to discover, it may be able to perfectly model a real-world suture. We have begun to find ourselves in a dilemma wondering if we have yet to discover the ideal parameters for the Colliders / Rigid Bodies / Physics Materials on each node and the Configurable Joints between them. Or have we reached the limitation of our this type of structure and need to find an entirely different model?

So far we have been operating from the assumption that we have not yet found the ideal parameters. We have, however, found that changes in these parameters do materially affect the suture's behavior and have been able to move in the direction of desired behavior. Currently, inexperienced users are occasionally able to use the virtual forceps to grasp the simulated suture and tie a knot in it. Knowing this it is more than likely that those with suturing experience will be able to consistently tie knots, though this still needs to be tested on the current model. In addition to this, while tying the knot and manipulating the suture under normal circumstances, the suture remains relatively stable whereas past models would have joints break and/or the suture would explode. This occurs when Unity's physics engine cannot find a way to bring two subsequent nodes back together and the entire suture flies off the screen trying to find the equilibrium point. Finally, the current model interacts well with the forceps that the user uses to manipulate the suture's position and rotation. Previous models, which were made more rigid by adjusting the spring force on the joint, would periodically rebound or bounce away from the forceps when the user attempted to grasp a portion of the suture. While this still can happen in the current model, the likelihood of it happening has decreased considerably and when it does happen, the rebound is typically not as noticeable or as strong as in past models.

While all the points made above are definitely a step in the right direction, the current model is still missing important characteristics of a real-world suture. Most notably, the model does not contain the rigidity and overall feel that a suture in the real world exhibits. An actual suture has certain characteristics and rigidity that is often compared to that of a human hair or fishing line. The current suture, with its lack of rigidity, behaves more like cooked spaghetti. Even though it is too floppy, this actually allows for an easier knot tying experience when compared to previous models. However, when a knot is tied, the suture does not stay in place and it starts to untangle itself. While this can be fixed by increasing the friction on the nodes' Physics Material, this makes it harder to tie a knot and makes it more likely to explode. In addition to this, there are a few bugs with the current model that break the immersion of a real suturing environment. As noted above, the model can explode while a real suture would simply break into two pieces when too much force is applied. Also, when the user grasps a node in between a joint, the Unity physics engine cannot find the equilibrium point for those joined nodes, since the forceps are in the way. This causes the suture to spasm until the user ungrasps it and makes it difficult to grasp the other side of the suture with the other forceps due to the spasm. In extreme circumstances, where the joints are under too much stress, this bug will cause the suture to explode. The last known bug, which is mentioned above, is the rebound bug, in which the suture moves away from the forceps while attempting to grasp the suture. While this bug has been minimized and its effects are mostly negligible in the current model, finding a solution to this bug and all others previously mentioned is necessary for creating an immersive suturing experience.

6 Parameter Variations and Observed Effects

During the process of tweaking parameters of various entities related to the suture, we began to document how changes to different parameters affect the suture's behavior and the positives and negatives that come with these changes. Oftentimes when tweaking parameters, the model felt more like a real suture, but it introduced new bugs or made existing bugs more common and/or had a worse effect when they occurred. Unfortunately, the opposite was also found to be true.

Figure 2 shows just a few of the parameters of a Configurable Joint, with the values that we found to work best. The most common parameter we tweaked was the Angular Y Limit and

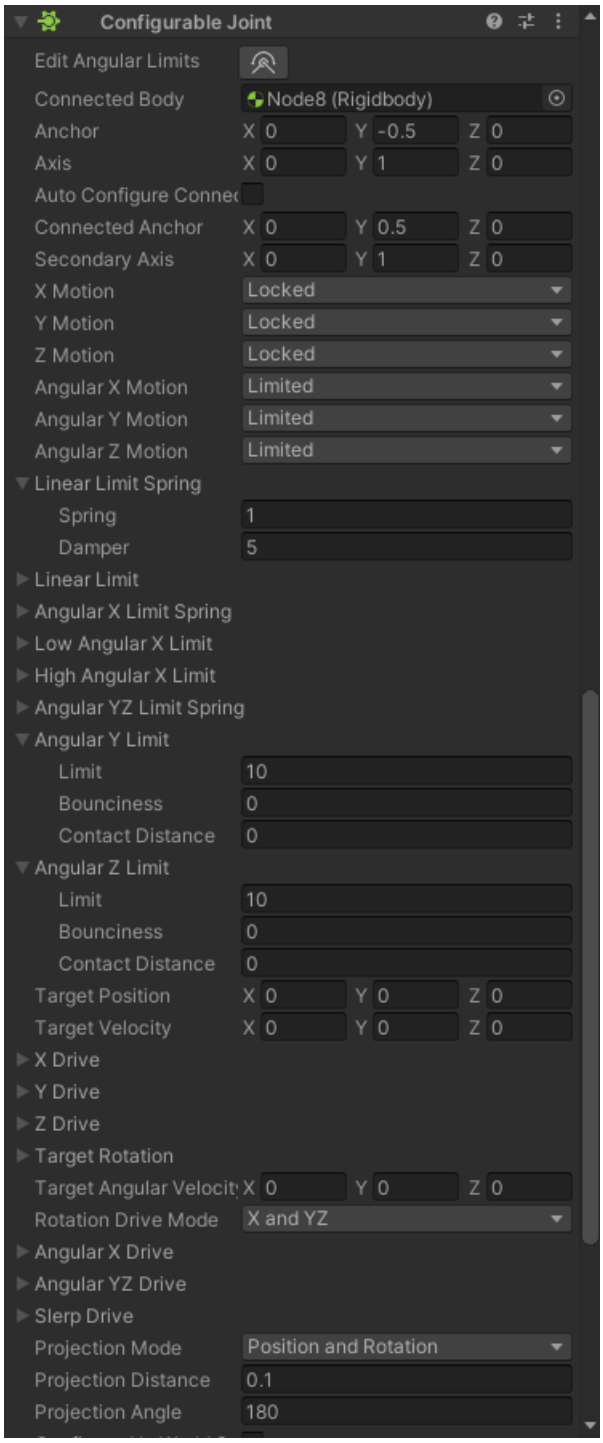

Figure 2: Configurable Joint parameters.

the Angular X Limit on the nodes' Configurable Joint. This value determined how far a joint was allowed to bend before the spring force would begin acting on the Rigid Bodies. One would think setting this parameter to 0 and the spring force to a very high value would make the joint stiff and not bend, but this was not the case. It did make the suture more rigid, but only to a certain degree that was not nearly as rigid as that of a real suture. With what we know, this was caused by some external force still being acted on the joint that Unity deemed stronger than the spring force. The external force is gravity acting on each node and the forces that the forceps apply on the suture while pulling on it. We ultimately set the angular limits to 10 degrees since too high of a value allowed the suture to be too flexible and anything less made the suture hard to bend, which introduced more stress on the joints. More stress created a higher likelihood of the suture exploding.

After we found an agreed upon value for the angular limits, we further experimented with the linear limit spring on the Configurable Joint. The associated value dictates the strength of the forces that the joint applies to the connected node when it leaves the angular limit. The spring value had to be set such that the spring was not too strong, so that there would not be a lot of stress on the joint when bending it and tying it in a knot, while at the same time the joint had to be rigid. With numerous testing, the value of 1 was found to be the best balance of the two. The damper value has been tricky to calibrate. This value softens the force of the spring to eliminate jittering. A high value completely removes the jittering effect, but makes the spring force far too weak. A low value does nothing to fix the jittering effect of the suture. Through testing we arrived at a value of 5, but this value may need to be further tested.

Figure 2 shows a variety of other parameters that can also be tweaked to perfect the joint. These values have been briefly experimented with and thus far have proved to have little to no effect compared to the parameters listed above or changed the joint in a way that was not desired. For this reason, these values will not be examined in detail. Furthermore, the Colliders, Rigid Bodies, and Physics Materials have parameters that can be adjusted as well. In future models, these parameters will be further tested.

7 Future Work

There are many other components to the simulator that need to be completed or improved before it will be ready for use by students. Focusing on the model of the suture for now, we have many parameters left to consider. We have also thought that it may be necessary for these parameters to change dynamically based on properties such as the number of other nodes in proximity or whether or not this particular node has been grasped. If we cannot achieve a sufficiently realistic model through any tweaking of these parameters we may abandon our nodes-and-joints model in favor of something else entirely, such as a deformable mesh.

8 Conclusion

Unity has proven itself to be an effective, but overwhelmingly flexible, system for modeling 3D phenomena. Modeling a microsurgical suture is especially difficult because it must balance flexibility and rigidity, distribute forces over its entire surface, and respond realistically to minute forces. We believe that a series of nodes connected through Configurable Joints is the most promising technique for building such a model, and have been able to fine-tune it by experimenting with many of the parameters that are provided by that Unity component.

References

[1] S. Ramachandran, A. M. Ghanem, S. R. Myers, Assessment of microsurgery competency – where are we now?, *Microsurgery*, *33*(5):(2013), 406–415.

[2] R. M. Studinger, M. M. Bradford, I. T. Jackson, Microsurgical training: is it adequate for the operating room?, *Eur. J. Plast. Surg.*, *28*(2):(2005), 91–93.

[3] A. M. Ghanem, N. Hachach-Haram, C. C. M. Leung, S. R. Myers, A systematic review of evidence for education and training interventions in microsurgery, *Arch. Plast. Surg.*, *40*(4):(2013), 312–319.

[4] C. C. M. Leung, A. M. Ghanem, P. Tos, M. Ionnac, S. Froschauer, S. R. Myers, Towards a global understanding and standardisation of education and training in microsurgery, *Arch. Plast. Surg.*, *40*(4):(2013), 304–311.

[5] M. Singh, N. Ziolkowski, S. Ramachandran, S. R. Myers, A. M. Ghanem, Development of a five-day microsurgery simulation training course: a cost analysis, *Arch. Plast. Surg.*, *41*(3):(2014), 213–217.

[6] D. Dumestre, J. K. Yeung, C. Temple-Oberle, Evidence-based microsurgical skills acquisition series part 2: validated assessment instruments – a systematic review, *J. Surg. Educ.*, *72*(1):(2015), 80–89.

[7] K.-S. Choi, S.-H. Chan, W.-M. Pang, Virtual suturing simulation based on commodity physics engine for medial learning, *J. Med. Syst.*, *36*(3):(2012), 1781–1793.

[8] S. Haque, S. Srinivasan, A meta-analysis of the training effectiveness of virtual reality surgical simulators, *IEEE Trans. Inf. Technol. Biomed.*, *10*(1):(2006), 51–58.

[9] H. Kazemi, J. K. Rappel, T. Poston, B. H. Lim, E. Burdet, C. L. Teo, Assessing suturing techniques using a virtual reality surgical simulator, *Microsurgery*, *30*(6):(2010), 479–486.

[10] I. Kozak, P. Banerjee, J. Luo, C. Luciano, Virtual reality simulator for vitreoretinal surgery using integrated OCT data, *Clin. Ophthalmol.*, *8*:(2014), 669–672.

[11] G. S. Ruthenbeck, K. J. Reynolds, Virtual reality surgical simulator software development tools, *J. Simul.*, *7*:(2013), 101–108.

[12] K. Triantafyllou, L. D. Lazaridis, G. D. Dimitriadis, Virtual reality simulators for gastrointestinal endoscopy training, *World J. Gastrointest. Endosc.*, *6*(1):(2014), 6–12.

[13] D. S.-C. Ng, Z. Sun, A. L. Young, S. T.-C. Ko, J. K.-H. Lok, T. Y.-Y. Lai, S. Sikder, C. C. Tham, Impact of virtual reality simulation on learning barriers of phacoemulsification perceived by residents, *Clin. Ophthalmol.*, *12*:(2018), 885–893.

[14] M. Radia, M. Arunakirinathan, D. Sibley, A guide to eyes: ophthalmic simulators, *Bull. R. Coll. Surg. Eng.*, *100*(4):(2018), 169–171.

[15] J. R. Broyles, P. Glick, J. Hu, Y.-W. Lim, Cataract blindness and simulation-based training for cataract surgeons, *Rand Health Q.*, *3*(1).

[16] A. Singh, G. H. Strauss, High-fidelity cataract surgery simulation and third world blindness, *Surg. Innov.*, *22*(2):(2015), 189–193.

[17] J. Zhang, Y. Lyu, Y. Wang, Y. Nie, X. Yang, J. Zhang, J. Cheng, Development of laparoscopic cholecystectomy simulator based on unity game engine, in *CVMP'18*, pages 1–9 (2018).

[18] K. Fan, A. Marzullo, N. Pasini, A. Rota, M. Pecorella, G. Ferrigno, E. D. Momi, A unity-based da vinci robot simulator for surgical training, in *BioRob'22*, pages 1–6 (2022).

[19] F. Zhang, Z. Sun, T. Wang, Brain modeling for surgical training on the basis of unity 3d, in *ISAIR'22*, pages 1–8 (2022).

[20] L. Khan, Y.-J. Choi, M. Hong, Cutting simulation in unity 3d using position based dynamics with various refinement levels, *Electronics*, *11*(14).

[21] *Unity User Manual*.