COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2013; **24**:25–41
Published online 2 May 2012 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cav.1434

RESEARCH ARTICLE

vKASS: a surgical procedure simulation system for arthroscopic anterior cruciate ligament reconstruction

Yanzhen Wang[1]*, Yueshan Xiong[2], Kai Xu[2] and Dong Liu[2]

[1] College of Information Systems and Management, National University of Defense Technology, 137 Yanwachi Street 410073 Changsha, Hunan, China
[2] School of Computer Science, National University of Defense Technology, 137 Yanwachi Street 410073 Changsha, Hunan, China

ABSTRACT

Arthroscopic surgeries, which are widely used for anterior cruciate ligament (ACL) reconstruction, not only require advanced hand–eye coordination but also involve complicated surgical procedure, necessitating simulation-based training for surgeons. This paper describes a surgical procedure simulation system for the training of arthroscopic ACL reconstruction. Different from existing simulation-based training systems for basic surgical skills, this system provides a complete simulation for the entire procedure of arthroscopic ACL reconstruction, involving operations such as puncturing, probing, incision, and drilling. In this system, we employ a linear elastic finite element method and position-based dynamics for deformable modeling. Simplified vertex duplicating method and an implementation of real-time Boolean operations are proposed for the topological change of tissue models involved in the incision simulation and tunnel construction. Two specially designed force feedback models are introduced for the haptic rendering of probing and drilling operations. By using these fast and stable simulation methods, this system is able to provide real-time realistic graphical and haptic feedback to the user, making it efficient and practical for training purpose. Feedback from the surgeon trainees shows that this system is very effective in training not only for basic surgical skills but also for the complex surgical procedure. Copyright © 2012 John Wiley & Sons, Ltd.

KEYWORDS
virtual surgery; arthroscopy; ACL reconstruction; procedure simulation; haptic feedback

***Correspondence**
Yanzhen Wang, College of Information Systems and Management, National University of Defense Technology, 137 Yanwachi Street 410073 Changsha, Hunan, China.
E-mail: yanzhen.wang@gmail.com

1. INTRODUCTION

Acting as one of the main stabilizing ligaments of the knee, the anterior cruciate ligament (ACL) plays a crucial role in preventing forward movement of the tibia from underneath the femur. ACL injuries, such as a torn ACL, happen when an extremely large burden is loaded on the knee when people do contact sports and those that involve fast changes in direction and twisting and pivoting movements of the knee. A torn ACL has serious implications for the stability and function of the knee joint. In a traditional open surgery for ACL reconstruction, the surgeon has to open the entire knee joint so as to gain access to the torn ACL, introducing very large trauma to the patient.

Nowadays, knee arthroscopy is widely used in repairing the torn ACL. It has several advantages over traditional open surgeries. Knee arthroscopy reduces surgical trauma and postoperative recovery time and introduces fewer complications. However, there are also some limitations on narrow view scope, nonintuitive perception and restricted instrument movement, and so on. Therefore, intensive training is needed for novice surgeons of knee arthroscopy before they can perform surgery on patients in a real-world setting.

Traditional methods using animals, cadavers, or plastic models are insufficient for intensive training purpose. The anatomy of animals is typically different from that of humans, and surgical training on animals raises some ethical concerns. Cadavers are expensive and relatively difficult to obtain. Moreover, both animals and cadavers typically cannot be reused. Plastic models are cheaper and reusable, but they cannot provide realistic visual and haptic feedback to trainees. Therefore, virtual reality (VR)-based training systems become a good choice for training of complex surgery skills. Surgery simulators are cost-efficient and theoretically can be reused infinite times. More

importantly, one can easily alter the virtual anatomy within the VR environment to simulate different cases (especially cases that are rare in real world). The versatility of surgery simulators facilitates the design of systematic training protocols. This technique largely drives the popularization of arthroscopic surgery.

1.1. Related Work

Since the last decade of the 20th century, many surgery simulators have been developed, either for research or for commercial purpose. In this section, we have a quick review only on those mostly related to ours.

Early knee arthroscopic surgery simulation/traing systems [1–4] mostly rely on high-end workstations for real-time visualization. Although key issues in 3D reconstruction from 2D medical images and simple 3D interactions are addressed, some of them lack haptic perception, especially force feedback. The work of Müller *et al.* [4] might implemented first approaches to the simulation of tissue deformation caused by collisions with the instruments. However, virtual anatomical models in these systems are relatively simple and the virtual surgery scenes are not visually realistic enough.

To provide haptic perception, Megali *et al.* [5] adopted a mock-up leg model for limited force feedback. The entire system provides multimedia modules to train and assess anatomical and procedural knowledge, as well as a VR-based simulator for training perceptual–motor skills. Although good cognitive learning can be achieved, structural anatomic deformation is not modeled in this system. The VR arthroscopic knee simulator system [6] developed by the American Academy of Orthopaedic Surgeons employs the PHANToM Desktop (SensAble Technologies, Inc. [STI], Woburn, MA, USA) haptic device [7] as the haptic feedback interface, which is the same as our system. However, the system adopts a volumetric representation for anatomic structures, whereas our system uses triangular and tetrahedral meshes. Volumetric representations are effective for static visualization of anatomical structures restored from medical images, but high-quality rendering of them is very computationally expensive. Therefore, the simulation system mentioned previously uses low-resolution volumetric models, and the visual effect is less than satisfactory. Heng *et al.* [8] developed a training system for virtual arthroscopic knee surgery. They used a tailor-made haptic device that can provide six degrees of freedom (DoF) of force feedback to the user.

The aforementioned surgery simulators mainly focus on training for arthroscopic diagnoses or basic skills needed for performing arthroscopic surgery. However, they cannot provide training for specific surgical procedures for several reasons. First, early arthroscopic techniques were mostly used for diagnoses rather than treatment. Second, specific surgical procedures are too complicated to simulate, given the software and hardware specification of early simulators. Recently, a commercial arthroscopy simulator called insightArthro VR was developed by GMV (Madrid, Spain) [9,10]. It has been widely used in European countries and is considered to be an efficient tool for arthroscopic surgery training. It can be learned from the product description that insightArthro VR can provide training for several surgical procedures including ACL reconstruction [11], but we could not obtain the technical details because it is a commercial system.

1.2. Contribution

We developed an advanced knee arthroscopic surgery training system named *vKASS*, which stands for *virtual Knee Arthroscopic Surgery Simulator*. vKASS provides training not only for basic surgery skills, such as triangulation and hand–eye coordination, but also for the entire complex ACL reconstruction procedure. The procedure simulated in our system is a standard method of ACL reconstruction, in which an EndoButton and a hamstring (semitendinous and gracilis) tendon autograft are used. The development of our system, from the modeling of anatomic structures to the design of the surgical procedure, is supervised by experts in orthopedics and arthroscopy. This contributes in providing a realistic multimodal feedback and a correct surgical procedure simulation to the surgeon trainees.

There are many issues that have to be taken into account in order to realistically simulate the complicated surgical procedure. These issues include graphical and haptic rendering for soft tissue deformation, incision, and drilling on anatomic structures, each of which is regarded to be computationally expensive in the literature. Moreover, they have to be nicely integrated into one system, in which they are processed simultaneously. Surgery simulation systems also share the common issues of concern with other VR systems, including efficiency, robustness, and precision. Bro-Nielsen regards real-time performance and stability to be the most crucial requirements for surgery simulation, and relatively less important, the precision [12]. We follow this proposition in the development of our system and hence always develop or adopt fast and robust methods in the simulation. Although these methods may not be comparable with other more advanced ones in terms of precision, they are very efficient and stable. Nevertheless, they are still able to provide physically plausible results sufficient to the purpose of surgical procedure training.

(1) Different deformable modeling methods are used to simulate various deformation behaviors of different objects. A linear elastic finite element method (FEM) is adopted for deformation simulation of soft tissues, such as skin and meniscuses. On the other hand, we propose a method based on position-based dynamics and generalized cylinders to model rope-like objects such as tendons, ligaments, and surgical sutures.
(2) For the incision simulation on deformable models, we propose a simplified vertex duplicating method that is able to provide realistic graphical rendering for real-time incision operations on surface meshes.

Comp. Anim. Virtual Worlds 2013; **24**:25–41 © 2012 John Wiley & Sons, Ltd.
DOI: 10.1002/cav

The method is simple to implement and efficient in computation. Moreover, it introduces much less modification to the mesh model than element subdivision methods and therefore facilitates the following update of the corresponding deformable model.

(3) Drilling operations on the bone can be implemented by using Boolean operations. However, the two models involved in the Boolean operation are ever-changing during the drilling simulation; hence, existing typical methods for Boolean operations have to perform space partition to accelerate the intersection detection in every rendering frame. They are insufficient for the real-time simulation purpose. Therefore, we propose a simplified method of real-time Boolean operations for drilling operation along a fixed line, which is very efficient in computation.

(4) To provide realistic force feedback for the probing operation on the ACL, we develop a simple and efficient algorithm, in which both the ACL and the probe are approximated using sequences of connected line segments. This simplified representation for the ACL is also consistent with its deformable model by using position-based dynamics with generalized cylinders.

(5) We designed a radial basis function (RBF)-based algorithm for the force feedback of drilling operations on the bone. It is efficient in computation and easy to implement and is able to provide stable and realistic haptic rendering to the user.

The remainder of this paper is structured as follows. In Section 2, we give an overview on the hardware and software architectures of vKASS, as well as the surgical procedure to be simulated. Key simulation techniques used in our system are described in Sections 3–5. Then, we describe some additional features that enhance the usability of vKASS to make it more suited for training purpose in Section 6. Finally, results of our system are presented in Section 7, and Section 8 concludes this paper.

2. SYSTEM OVERVIEW

2.1. System Architecture

The hardware platform of our system includes two STI PHANToM® Desktop haptic devices, which are placed inside a plastic leg model, and a generic personal computer. As shown in Figure 1, the entire system is placed on a movable cart. Currently, vKASS is running on a personal computer with Intel® Pentium® Dual Core E5200 2.5-GHz CPU (Intel Corp., Santa Clara, CA, USA), 4-GB main memory, and an NVIDIA GeForce 9600 GT graphics card (NVIDIA, Santa Clara, CA, USA). By using two haptic devices, our system provides the user with a two-hand haptic interface, in which the PHANToM® Desktop on the left side controls the movement of the arthroscope and the one on the right side controls the movement of other surgical instruments in use.

Figure 1. The hardware platform of our system.

We adopt C++ with the OpenGL library as the programming language. The GHOST® SDK [13] provided by STI is used to support the haptic devices. Figure 2 shows the software architecture of our system. The entire system includes a preprocessing phase and a real-time surgery simulation phase. In the preprocessing phase, geometrical models of soft tissues and surgical instruments are generated as surface (triangular) or volumetric (tetrahedral) meshes. In the real-time surgery simulation phase, the entire surgical procedure is simulated and is interactive for users. The minimal update rate needed for stable and pleasing haptic perception is much higher than that needed for satisfactory visual perception; hence, the real-time simulation runs on two separated processes according to the GHOST® SDK. In one of the two processes, which is called the *haptic process* or the *servo control loop* in GHOST® SDK, feedback force is computed and generated at a rate of $\sim$1 kHz. Other simulation computations, such as collision detection, soft tissue deformation, incision simulation, and visual rendering are performed in the other process called the *application process*, at a frame rate of $\sim$30 Hz.

2.2. Surgical Procedure

In our system, we implement the procedure simulation for arthroscopic ACL reconstruction surgery by using an EndoButton and a hamstring tendon autograft, which is one of the most popular methods followed by surgeons

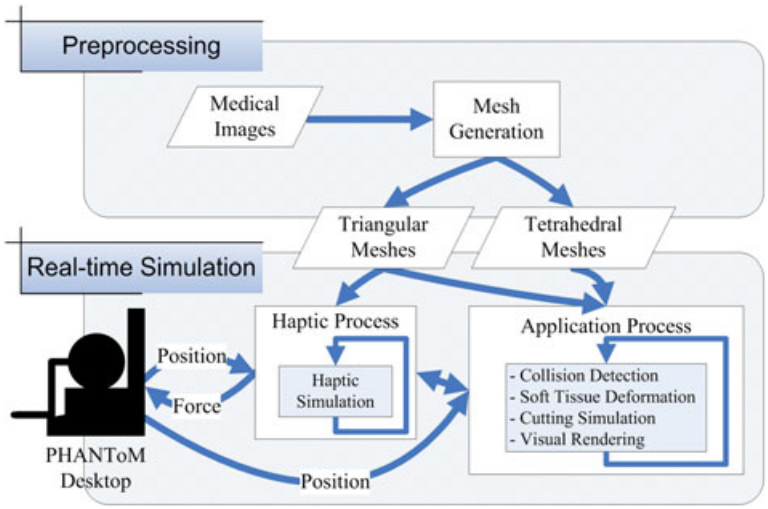


Figure 2. The software architecture of our system.

and is considered to be a standard. The entire procedure of such arthroscopic ACL reconstruction method is very complicated. In order to make the simulation tractable, we simplified the procedure by extracting the most crucial operations and ignoring some trivial ones. We further divide the procedure to be simulated into six stages:

(1) *Puncture stage.* Suppose the patient has a torn ACL in her or his right knee joint. At this stage, the surgeon selects appropriate positions for arthroscopic portals on the patient's leg and makes orifices in these positions by using a scalpel. Then, the arthroscope and the instrument to be used are introduced into the knee joint through the orifices on the left and right sides of the patient's knee, respectively.
(2) *Inspection and clean-up stage.* After the arthroscope is introduced into the knee joint, the surgeon would perform inspection on the torn ACL to evaluate the trauma. A hook-shaped probe will also be used to estimate the tension of the torn ACL. The result of inspection is very crucial for the choice of different treatments. Then, the surgeon would do some clean-up within the knee joint to make the surgery region clear. If the ACL reconstruction is chosen as the treatment, instruments such as ablators and clamps are used to cut and remove the entire torn ACL in this stage.
(3) *Graft generation stage.* The surgeon makes an opening on the patient's leg with a scalpel and fetches the semitendinous and gracilis tendons from the opening. Then, the autograft, which will be used to replace the torn ACL, is generated by using the semitendinous and gracilis tendons, as well as an EndoButton and some sutures.
(4) *Tunnel construction stage.* In this stage, two tunnels are constructed on the tibia and femur of the patient. It is very important that the position and the direction of the tunnels are correctly chosen, because it will affect the functional of the reconstructed ACL. Therefore, tibia and femur tunnel locators are used to fix the tunnel portals and to ensure that the drill proceeds in the desired direction.
(5) *Graft placing stage.* The generated graft is placed in the patient's knee joint through the tunnels. First, a leading acus with a piece of leading thread attached to its end is inserted into the knee joint through the instrument portal on the right and is then pulled out through the femur tunnel, with the end of the leading thread left visible under the arthroscopy in the knee joint. Second, the end of the leading thread that stays in the joint is pulled out by a clamp through the tibia tunnel. Third, the autograft is attached to the end of the leading thread and is pulled into the joint by the leading thread through the tibia tunnel. At last, the EndoButton is used to fix the upper end of the autograft to the femur.
(6) *Accomplishment stage.* In this stage, the surgeon bends and straightens the patient's knee joint 20–30 times to make the autograft tight. And then, the lower end of the autograft is fixed to the tibia at the tibia tunnel entry by using an interface screw.

As depicted in Figure 3, a sequence of states is used to simulate the simplified surgical procedure in our system. In the figure, the important techniques involved in the simulation of each procedure stage are also listed in the corresponding state node, in the form of the titles of the sections describing these techniques. When the user has completed all the key operations in one stage, she or he can move on to the next stage by pressing a trigger button.

3. DATA PREPROCESSING

For geometrical modeling of anatomic structures in the knee joint, we simply take the typical routine that is widely used in previous surgery simulators [8]. A sequence of

DOI: 10.1002/cav

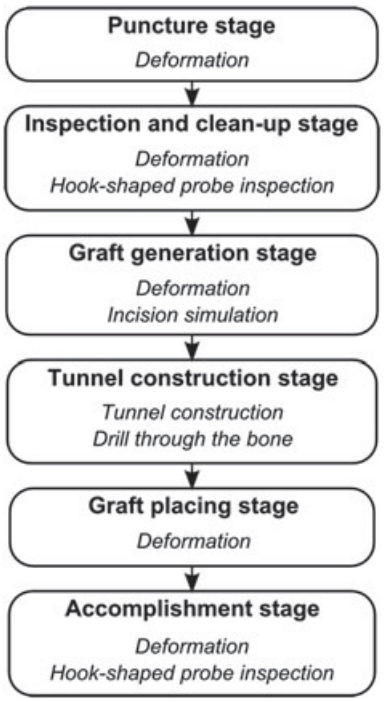


Figure 3. The simplified surgical procedure.

colored medical images corresponding to the knee joint region in the Visible Human Project data set [14] is chosen to be the input of the preprocessing phase of our system. First, segmentation for these images is performed. Then, the boundaries of the tissues of interest are extracted in every image by using the active contour model [15]. Finally, with the use of the Nuages [16] software, triangular or tetrahedral mesh models of the tissues are generated from the sequences of contours. Although this method is very effective, the surface or volumetric mesh models directly reconstructed from medical images suffer from several problems. Because of the noise existing in the medical images, the surface of the mesh models may not be smooth enough for haptic or visual rendering or the shape of the surface may not conform to the real tissue. Moreover, restricted by the tetrahedral mesh reconstruction method, the quality of the tetrahedral elements in the resulting meshes is not sufficient for numerical computations such as deformable modeling. Therefore, we perform a tetrahedral mesh optimization and an interactive triangular mesh editing to obtain mesh models with high-quality elements and desirable shapes before they are fed in the real-time surgery simulation stage.

3.1. Mesh Optimization

The quality of the elements in a tetrahedral mesh is crucial for the numerical accuracy and convergence of deformable modeling using FEMs or mass–spring systems. On the other hand, the quality of surface elements also affects the visual appearance of the mesh model. However, tetrahedral meshes generated from cross-sectional data by using the Nuages software typically do not possess the desired quality. Thus, we employ a two-stage method to obtain tetrahedral meshes with both smooth surfaces and well-shaped interior elements. In the first stage, feature-preserving surface fairing, such as the $\lambda|\mu$ method [17], is performed to reduce the noise on the surface. In the second stage, we utilize a variational tetrahedral mesh optimization method based on quality encoding [18] to improve the quality of the interior elements. The resulting tetrahedral meshes after optimization have better dihedral angles than the original ones. Moreover, our optimization method also significantly improves the quality of boundary elements through a harmonic-guided optimization.

3.2. Online Mesh Editing

Because of the noise in the original medical images or the error introduced by the image segmentation and mesh generation, the shape of the mesh models may not exactly reveal the actual anatomy. Even though the aforementioned optimization method is able to reduce the noise and generate smooth surfaces, the correct shape of the anatomy cannot be retrieved. Therefore, we provide an interactive online mesh editing module within our system. Experts in orthopedics and anatomy can modify the surface meshes of the tissue models in the surgery scene interactively. Thanks to the PHANToM® Desktop haptic device, the modification of the anatomy models can be performed directly in 3D space. Thus, the control of the shape in our online mesh editing is more intuitive and effective than other 3D modeling softwares, such as 3ds Max® and Maya® (Autodesk, Inc., San Rafael, CA, USA). To modify the surface mesh, the user uses the stylus of the haptic device to control a cursor in the surgery scene to grab a vertex on the model surface and move it to a desired place. Other vertices in the neighborhood of the grabbed vertex are also displaced according to the displacement of the grabbed vertex and a weight decided by a soft selection scheme, which is the same as in 3ds Max®, in order to produce a smooth surface after the editing. The falloff of the soft selection can also be adjusted by the user. Further exploiting the ability of the haptic device, a resistant force with its magnitude proportional to the norm of the displacement of the grabbed vertex is exerted on the stylus of the haptic device, in order to provide the user with an appropriate interactive feedback to make the editing operations more accurate.

4. GRAPHICAL FEEDBACK

4.1. Deformation

There are many deformable objects existing in the virtual surgery scene, and typical deformable modeling methods are notoriously expensive in computation. Therefore, deformable modeling is one of the most important issues in surgery simulation. Particularly, in our system, there are

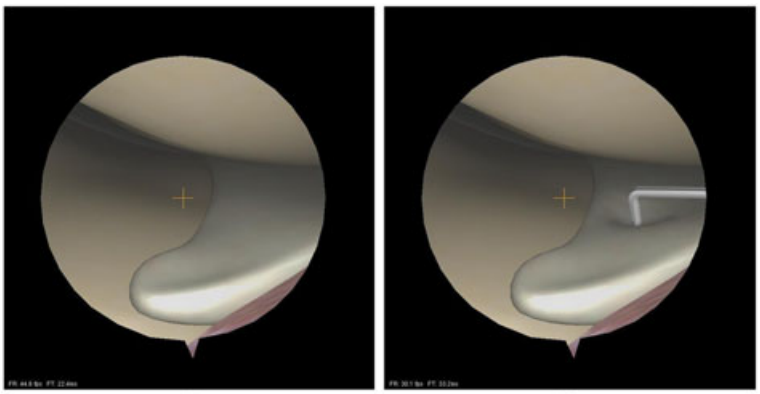


Figure 4. Deformation of meniscus: (a) undeformed meniscus and (b) meniscus deformed by a hook-shaped probe.

soft tissues, such as skin outside the knee joint, meniscuses, tendons, and cruciate ligaments, as well as some rope-like surgical instruments, such as sutures, that all need to be simulated as deformable objects. Keeping in mind that the most important requirements in VR systems are the real-time performance and the stability, we choose to use efficient and simple deformable modeling methods. Unfortunately, the deformable objects in our system vary much in their deformation behavior, such as the visoelasticity of soft tissues and the insensibility of sutures; therefore, there is no universal method that can provide both efficiency and realistic deformation result for such a wide spectrum of various behaviors. Thus, we use different deformable modeling methods for different objects. Because our system is designed for surgical procedure simulation, the physical and numerical accuracy of the deformation simulation is not as important as its real-time performance. Nevertheless, the result of the simulation is still physically and visually plausible.

4.1.1. Linear elastic finite element method.

For deformation modeling of soft tissues such as skin and meniscuses, a common choice followed by most of the existing surgery simulation systems is to use FEMs. Nonlinear FEMs are accurate and sufficient for large-scale deformation, but the computation cost becomes too heavy for a real-time system as the data set grows larger. On the other hand, linear elastic FEMs are more efficient in computation, but they cannot provide precise simulation for large-scale nonlinear deformation. Another choice for deformable modeling is to use mass–spring systems. Although mass–spring systems are very efficient, their accuracy is typically lower than FEMs, and the result is dependent on the discretization (the vertex distribution as well as the connectivity among the vertices in the mesh) of the object. We noticed that the soft tissues involved in an arthroscopic surgery mainly undergo very small-scale deformation, for the reason that arthroscopy is a type of minimal invasive surgery. Therefore, in order to guarantee the real-time performance of our system, we use a highly simplified linear elastic FEM that is the same as the method proposed in [12]. Figure 4 shows the deformation on meniscus simulated by our system by using the linear elastic FEM.

4.1.2. Position-based dynamics with generalized cylinders.

Rope-like objects commonly undergo large nonlinear deformation of their medial axes but keep their length almost unchanged during interaction. Therefore, the deformation behavior of these objects is hard to simulate by using simple linear elastic FEMs or mass–spring systems. We are encountered with many rope-like objects, such as tendons, ligaments, and various kinds of sutures in the simulation for arthroscopic ACL reconstruction. To model their deformation behaviors efficiently, we utilize the position-based dynamics method [19] combined with generalized cylinders [20] for deformable modeling.

As depicted in Figure 5, the surface mesh model of a rope-like object is remeshed so that each vertex in the new surface mesh is located on one of the N cross sections, which are perpendicular to the medial axis of the object. The medial axis is also discretized and represented by N nodes L_i $(1 \leq i \leq N)$, each placed at the central point of

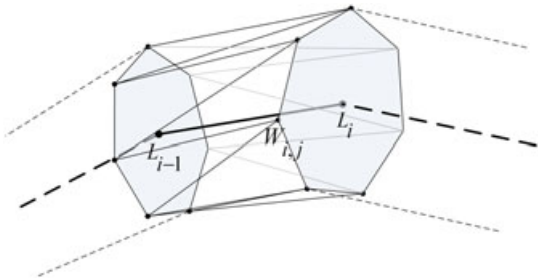


Figure 5. The generalized cylinder model for rope-like objects.

DOI: 10.1002/cav

the ith cross section. Initially, $\vec{w}_{ij}^{0}$, the local coordinate of each vertex j on the ith cross section with respect to L_i is computed and stored. During user interaction, the deformation of the medial axis can be computed very efficiently by performing position-based dynamics on the consecutively linked nodes L_i. Then, the new position of each vertex j on the ith cross section, $\vec{w}_{ij}$, can be obtained by performing correct transformation on $\vec{w}_{ij}^{0}$:

$$\vec{w}_{ij} = \vec{t}_i + \vec{R}_i \vec{w}_{ij}^{0} \tag{1}$$

where $\vec{t}_i$ and $\vec{R}_i$ are the translation vector and rotation matrix of the ith cross section, respectively. For $\vec{t}_i$, we simply use the current position of L_i. And $\vec{R}_i$ can be derived from the deviation between the original and current positions of all the cross sections. We then use the new positions of the vertices in the mesh to render the deformed rope-like object. Figure 6 shows an example of a deformed rope-like object in our system.

4.2. Incision Simulation

In the *graft generation stage* of the surgical procedure, the surgeon makes an opening on the skin outside the knee joint in order to fetch the semitendinous and gracilis tendons. Thus, we need to perform an incision simulation on the deformable model of skin. Typically, incision simulations are implemented by using either element removal or element subdivision in the literature. Element removal methods are efficient and easy to implement, and the modification of the corresponding deformable model can also be achieved efficiently. However, the result of element removal methods is not precise and not visually pleasing. On the other hand, element subdivision methods are more accurate, but they will introduce much more new elements and topological changes of the mesh into the system. Therefore, the modification of the corresponding deformable model is not straightforward, and the efficiency of the real-time simulation will be harmed.

For the incision simulation on the skin, we present a simplified vertex duplication method for surface meshes as shown in Figure 7. Firstly, we record the cutting plane (line segment AB in Figure 7 gives a 2D example) swept by the scalpel blade and find the intersected edges in the surface mesh model. Secondly, for each intersected edge, the closest vertex to the cutting plane is moved to the intersection point between this edge and the cutting plane. Then, each one of the displaced vertices is duplicated, and the duplicated vertices are assigned to the triangles on the two sides of the cutting plane. Finally, we generate the triangles in the groove introduced by the cutting operation. With the use of this method, the incision on the skin surface can be simulated efficiently with the resulting groove conforming to the cutting plane. Moreover, our simplified vertex duplication method introduces only a few number of extra vertices and elements into the system, whereas most of the original elements and vertices are kept unchanged. Therefore, the stiffness matrix used in the FEM computation can be updated with no difficulty, and the efficiency of the deformable modeling will not be affected. The result of the incision simulation method in our system is shown in

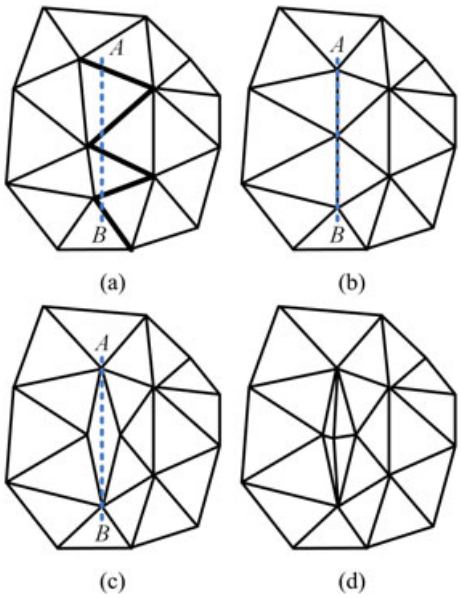


Figure 7. Simplified vertex duplicating method for the incision simulation: (a) intersected edges, (b) vertex displacement, (c) vertex duplication, and (d) groove triangles generation.

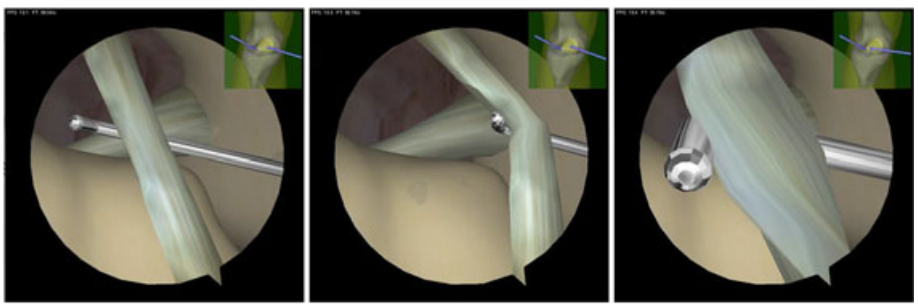


Figure 6. Deformation of the anterior cruciate ligament probed by a hook-shaped probe.

Figure 8. Note that we use a different texture for the groove triangles from other surface triangles of the skin to get a better rendering result.

4.3. Tunnel Construction

In the *tunnel construction stage*, the surgeon drills through the tibia and the femur to construct tunnels through which the autograft is going to be inserted into the knee joint and fixed in the proper position to act as the new ACL. We present an improved Boolean operation method for real-time drilling operations on surface mesh models. As shown in Figure 9, after the tunnel locator is placed, the direction and position of the tunnel to be constructed is decided by the line provided by the locator (e.g., line AB in the figure). Thus, the bone with the constructed tunnel can be represented by the result of Boolean operation $\Omega - C$, where Ω is the space bounded by the surface mesh model of the bone and C the space occupied by the drill head, which can be approximately represented by a cylinder along the drill direction with the radius of the drill head.

To dynamically render the process of the tunnel construction, a natural choice is performing the difference operation $\Omega - C$ once a rendering frame between Ω and C at each time step. Almost all the existing methods for computing Boolean operations between 3D mesh models consist of two steps: (1) *intersection detection* between the given meshes and (2) *reconstruction* of the resulting mesh from the intersection results of the first step. Because intersection detection between two meshes is computationally expensive, existing methods for Boolean operations mainly rely on some type of space partition, such as octree [21], which is performed in a preprocessing stage to accelerate this step. However, because the drill head proceeds along the drill direction as the drilling process goes on, the dynamic simulation for the drilling operation in vKASS needs to perform the Boolean operation between two ever-changing mesh models at a visual frame rate of $\sim$30 Hz. Therefore, the space partition has to be performed in each frame and hence loses the efficiency benefit of the "preprocessing." Moreover, to maintain the correct topology of the resulting mesh model, many issues have to be considered in the reconstruction step and therefore added to the computational cost. As a result, we designed an efficient method

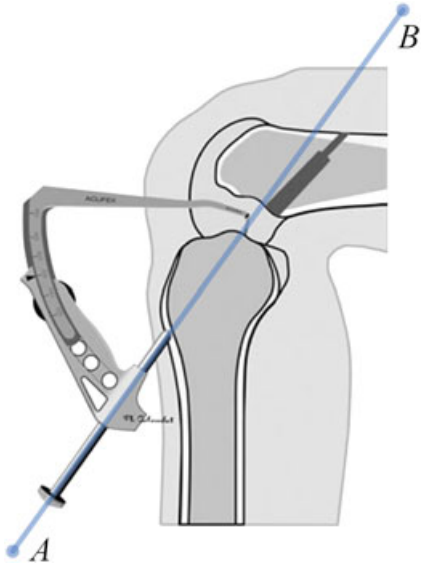


Figure 9. Locating of the tibia tunnel.

for the dynamic simulation for the tunnel construction process, making use of the prior knowledge that the drill will proceed along the line predefined by the locator.

Suppose that the boundary of Ω is represented by a triangular mesh $M = (V, E, F)$, where V, E, and F are the sets of mesh vertices, edges, and facets, respectively. If the direction along which the drill will proceed is given by a vector $\vec{d}$, then the set F can be divided into two subsets, the set of the front side facets $F_1 = \{f_i | f_i \in F \text{ and } \vec{f_i} \cdot \vec{d} \leq 0\}$ and the set of the back side facets $F_2 = \{f_i | f_i \in F \text{ and } \vec{f_i} \cdot \vec{d} > 0\}$, where $\vec{f_i}$ is the normal of the facet f_i. To achieve real-time dynamic simulation of the drilling operation, the generation of the tunnel is mainly performed in a preprocessing stage after the locator has been placed. Within the preprocessing stage, the upper bottom and the lower bottom of the tunnel are precomputed by the algorithm shown in Algorithm 1, where sets U and B, as well as lists L_1 and L_2, are all empty initially and N is the given maximum iteration times.

After the preprocessing stage, facets in the upper bottom and the lower bottom of the tunnel are stored in two sets U

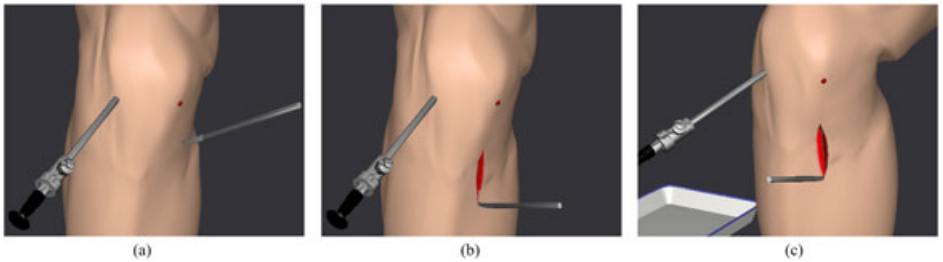


Figure 8. Result of the incision simulation on the leg: (a) the leg before incision and (b,c) results of the incision simulation from different viewpoints.

Algorithm 1: Algorithm for the preprocessing of the tunnel construction.

forall the $e_i \in E$ **do** // 1. Edge alignment.

  if e_i *intersects with the boundary of* C **then**

    $v =$ the intersection point;

    move the vertex of e_i, which is the closest to v, to the position of v;

forall the $f_i \in F_1$*(or* F_2*)* **do** // 2. Intersection test.

  if f_i *is totally inside* C **then**

    add f_i in set U (or B);

  else

    if f_i *intersects with the boundary of* C **then**

      insert f_i into list L_1 (or L_2);

for $k = 1 \ldots N$ **do** // 3. Iterative subdivision.

  forall the $f_i \in L_1$*(or* L_2*)* **do**

    $e =$ the edge of f_i which is totally inside C;

    subdivide f_i into two facets f_{i1} and f_{i2} by adding a vertex v' at the middle

    point of e;

    project v' on the boundary of C;

    subdivide the facets neighboring to e to keep the consistency of the mesh;

and B, respectively. The shape of the tunnel boundary of the upper bottom and the lower bottom on the surface becomes better as the iterative subdivision continues, whereas the shape of the newly introduced triangles becomes more poorly shaped. However, the surface mesh model of the bone is treated as rigid and does not participate in any numerical computation; thus, the quality of the triangles is not an issue. A 2D illustration of the preprocessing stage for the tunnel construction is shown in Figure 10.

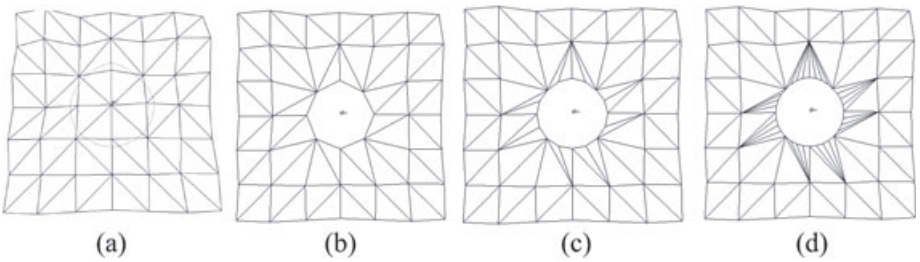


Figure 10. Boolean operation on the surface mesh: (a) the surface mesh and the boundary of the tunnel and (b) edge alignment and intersection test, iterative subdivision, and vertex displacement after (c) one iteration and (d) two iterations.

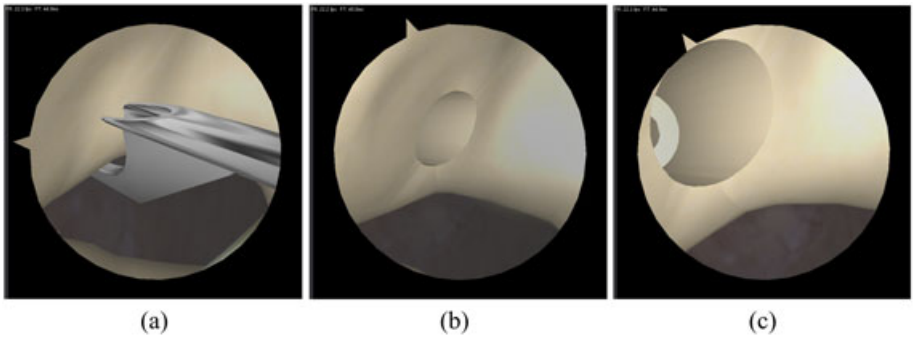


Figure 11. Results of the femur tunnel construction: (a) femur before drill and (b,c) drilled femur viewed from different points.

When the user starts the drilling operation, we remove the triangles within the upper bottom of the tunnel and render the lower bottom according to the current position of the drill head in each time step. The inner wall of the tunnel is rendered by simply using an advancing front method between the vertices on the edges of the upper bottom and the lower bottom. As soon as the bone is drilled through, the triangles of the lower bottom are removed, and we get the final result of the drilling operation. Figure 11 shows the result of the construction of the femur tunnel. By using this real-time Boolean operation method, we avoid performing space partition in every time step and almost all the computations for tunnel construction are precomputed in the preprocessing stage. Therefore, the drilling operation can be simulated very efficiently in real time.

5. HAPTIC FEEDBACK

Tactile feeling is very important for an arthroscopic surgeon to perform the surgical procedure, because the instrument operation is restricted by the small arthroscopic portals and is not directly observed by the eyes. Therefore, we employ PHANToM Desktop haptic devices and the accompanying GHOST SDK to provide realistic force feedback for the user. For force feedback produced by instrument–surface contact, we use the two basic models provided by GHOST SDK directly [13]. One of them is called the mass–spring–damper model, which is used for the generation of the force normal to the contact surface. The other one is called the stick-slip model, which is used to compute the force tangential to the contact surface, that is, static and dynamic frictions. To obtain a tactile feeling with high reality, we perform a trial-and-error procedure to select the proper haptic parameters for different tissue models, under the guidance of surgeons who have plenty of experience in knee arthroscopy.

In our surgical procedure simulation system, there exist other types of force feedback that cannot be modeled by the two basic force feedback models provided by GHOST SDK directly. Therefore, we present methods to generate such forces on the basis of the GHOST SDK models.

5.1. Hook-shaped Probe Inspection

In the *inspection and clean-up stage* and the *accomplishment stage*, the surgeon may use a hook-shaped probe to probe the torn ACL and reconstructed ACL respectively, in order to evaluate the tension of the ACL. For such operations, we present a simple method to compute the force exerted by the ACL on the probe. As depicted in Figure 12, both the ACL and the hook-shaped probe are represented by sequences of linked line segments. For ACL, this representation is consistent with the deformation simulation method presented in Section 4.1. For the probe, its simplified haptic model consists of two linked line segments AB and AC. The position of the probe in the previous time step is recorded by line segments $A'B'$ and $A'C'$, then the

DOI: 10.1002/cav

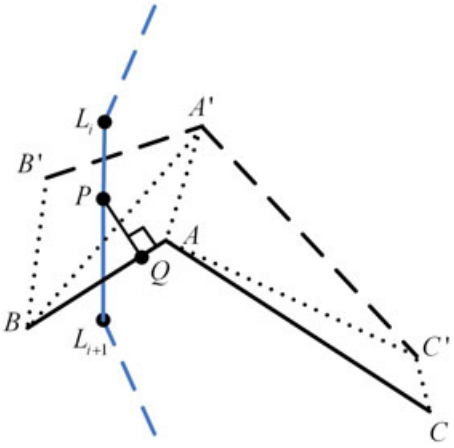


Figure 12. Force feedback model for hook-shaped probe inspection.

surface swept by the probe can be approximated by four triagles $\Delta BB'A', \Delta BA'A, \Delta AA'C$, and $\Delta AC'C$. The force on the probe is computed by using the algorithm described in Algorithm 2, where k_{stiff} is the parameter reflecting the stiffness of the ACL.

5.2. Drill through the bone

In the *tunnel construction stage*, the surgeon uses different types of drills to drill through the tibia and femur. The visual rendering of the tunnel construction has been described in Section 4.3, whereas the haptic rendering for the resistant force during the drill process is presented in this section. Because the direction and the position of the tunnel is fixed by the locator before the drilling operation, the tunnel to be constructed can be represented by a line segment, for example, AB in Figure 13, which has a total length L_{tunnel} and a normalized direction $\vec{d} = \overrightarrow{AB}/|AB|$. The current position of the drill head can be simply represented by a point as point P in Figure 13, and it is projected on AB to get the current drill depth D_{drill}. The

Algorithm 2: Algorithm for computing the probe inspection force.

for $i = 1 \ldots N$ **do**

perform intersection test between L_iL_{i+1} and $\Delta BB'A'/\Delta BA'A/\Delta AA'C/\Delta AC'C$;

if *there's an intersection* **then**

get the intersection point P;

if P *lies in* $\Delta BB'A'$ *or* $\Delta BA'A$ **then**

project P on AB to get the projection Q;

else

project P on AC to get the projection Q;

$\vec{f} = k_{stiff}\frac{\overrightarrow{QP}}{|QP|}$;

else

$\vec{f} = 0$;

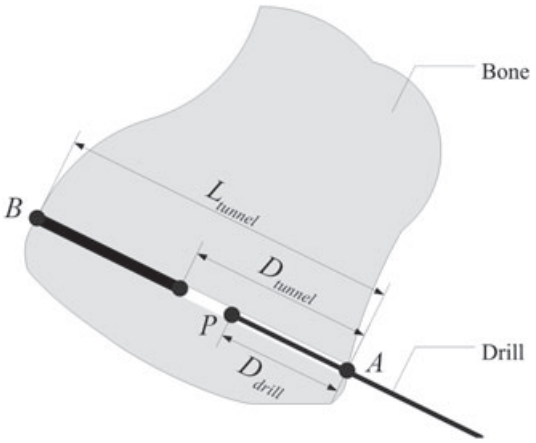


Figure 13. Illustration of the drill operation.

deepest position that has been achieved by the drill so far is recorded by the current tunnel depth D_{tunnel}. Then, the force feedback can be represented by the following piecewise function.

$$\vec{f}(D_{\text{drill}}) = \begin{cases} -F_{\text{const}}\vec{d} & \text{if } D_{\text{drill}} > D_{\text{tunnel}} \\ & \text{and } D_{\text{drill}} < L_{\text{tunnel}} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where $\vec{f}$ is the feedback force and F_{const} is a parameter that can be adjusted to control the magnitude of the force feedback. To calculate the resistant force exerting on the drill head, a direct and natural method is described in Algorithm 3. However, although it is very simple and efficient, this method produces undesirable vibrations in practice. This is because, that as the drill proceeds, not only the D_{drill} but also the D_{tunnel} changes.

Therefore, in our system, we present a method by using RBF to get a stable resistant force. After the direction and the position is fixed by the locator, the line segment representing the tunnel is evenly discretized into N (we take $N = 20$ in our system) vertices as shown in Figure 14. Their positions measured by the depth in the tunnel are $c_i = (i-1)(L_{\text{tunnel}}/(N-1)), (i = 1, 2, \ldots, N)$. As depicted in Figure 14, each vertex except the first one and the last one is assigned a piecewise linear RBF

$$\vec{\phi}(r_i) = \begin{cases} -(\frac{L_{\text{tunnel}}}{N-1} - r_i)F_{\text{const}}\vec{d} & \text{if } r_i < \frac{L_{\text{tunnel}}}{N-1} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where $r_i = \|D_{\text{drill}} - c_i\|$ is the distance between the drill head P and vertex i $(2 \le i \le N-1)$. At the beginning of the drill operation, all the vertices are marked as "active." As the drill proceeds, if any vertex i satisfies $D_{\text{drill}} - c_i > (L_{\text{tunnel}}/(N-1))$, then it is considered to be drilled off and set to be "inactive." Then, the feedback resistant force can be approximated by summing up the

Algorithm 3: A simple algorithm for computing the drill resistant force.

$D_{tunnel} = 0$;

while $true$ **do**

  compute D_{drill};

  if $D_{drill} > D_{tunnel}$ *and* $D_{drill} < L_{tunnel}$ **then**

    $\vec{f} = -F_{const}\vec{d}$;

    $D_{tunnel} = D_{drill}$;

  else

    $\vec{f} = 0$;

DOI: 10.1002/cav

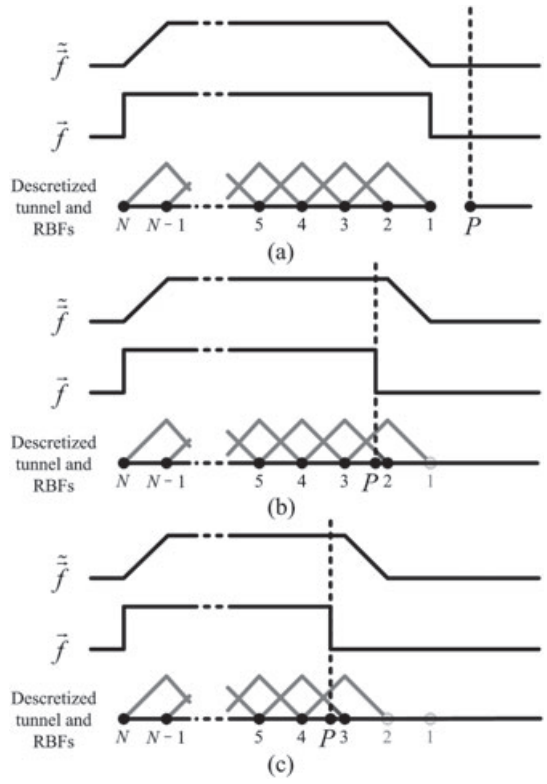


Figure 14. Process of resistance force simulation using radial basis functions (RBFs). The "active" vertices are depicted by solid dots and the "inactive" ones by empty circles.

RBFs of all the "active" vertices:

$$\vec{f}(D_{\text{drill}}) \approx \tilde{\vec{f}}(D_{\text{drill}}) = \sum_{\text{all the "active" vertices}} \vec{\phi}(r_i) \quad (4)$$

This approximation can be justified intuitively by the example shown in Figure 14, and the application in our system results in a realistic and stable resistant force.

6. ADDITIONAL FEATURES

In order to make our system more suited for computer-aided training, besides the surgical procedure simulation, we provide several additional functions in our system.

6.1. Panorama View

For novice surgeons to the arthroscopic surgery, it is usually hard to be used to the nonintuitive hand–eye coordination. They even can hardly figure out which part within the knee joint the arthroscope has reached or where the surgical instrument is. Therefore, we provide a navigation window in the software interface of our system (as shown in Figure 15), which provide a panorama view

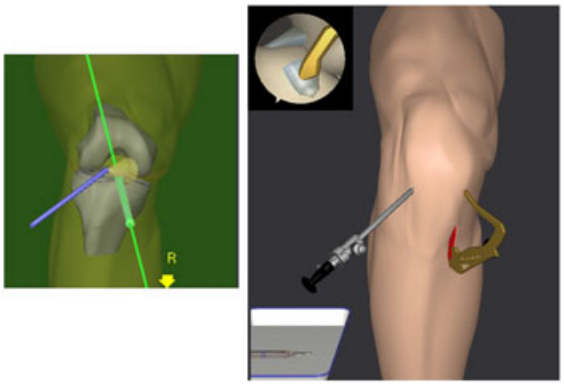


Figure 15. Panorama view and the corresponding surgery scene.

of the surgery scene. Within this window, tissues including bones and meniscuses, as well as the profile of the leg, are rendered in a transparent manner. The position and direction of the virtual arthroscope and the instrument in use are also shown in the window, and the current visible scope of the arthroscope is highlighted by a transparent yellow cone. During and after the *tunnel construction stage*, the direction of the tunnel to be drilled is shown by a green line segment, whereas the constructed tunnel is depicted as a green cylinder. The user can alter the viewpoint of the navigation window among three orthogonal directions: front, top, and left. This assistant viewport largely helps trainees in being used to the perception of positions and directions through the arthroscope and makes our system more suitable for novice surgeons. Of course, experienced trainees can easily shut down this panorama viewport.

6.2. Screen Recording

Our system provides a screen recording function to the user, so that the operations of an arthroscopic expert during the surgical procedure can be recorded and then replayed to the novice trainees for a reference. Moreover, the novice trainees can also record their own operations during the training. Then, they can play back the video file and learn from the successes, as well as the failures in their operations.

6.3. Undo the Surgical Procedure

The procedure of the arthroscopic ACL reconstruction is really complex; thus, trainees of our system are prone to make mistakes during their operations. When the trainee realizes that a mistake has been made but does not want to restart the entire training process, the undo mechanics ubiquitous in nowadays softwares will be very helpful. Therefore, we implemented an undo function in our system. A natural option is to maintain a stack of operations that have been performed by the user. This will allow the user to return to any previous state during one session of

simulation. However, the undo operation in our system is restricted to several restoration points, which are crucial to the entire surgical procedure, instead of any arbitrary states. We do this not only for the consideration of the system efficiency but also for the fact that most of the states in the surgical procedure are trivial and can be totally decided by several crucial states. For instance, if the user has placed the tibia tunnel locator, then the following drill operation will not change the position and direction of the final tibia tunnel. Hence, if the tibia tunnel has been constructed, it is meaningless for the user to redo the drilling operation. However, the user may want to go back to the state right before the placement of the tibia tunnel locator, in order to adjust the direction and the position of the tunnel. With these considerations, we provide the following six restoration points in our system.

(1) *Initial state.* This is the initial state of the surgical procedure, with no operation performed so far. Users may return to this restore point to redo the puncture operation, in order to correct the positions of the arthroscopic portals.
(2) *Inspection and clean-up stage.* At this point, the arthroscope has been introduced into the knee joint, and the portal for the instrument has been made. Users may return to this restore point to redo the training of inspection and surgical region clean-up operations.
(3) *Incision on the surface.* At this point, the user needs to use a scalpel to make an opening on the surface of the skin, in order to fetch the semitendinous and gracilis tendons for generation of the graft. If the position of the current incision is incorrect, the user can cancel it by going back to this point.
(4) *Tibia tunnel locating.* At this point, the user needs to properly place the tibia tunnel locator, in order to ensure the correct position and direction of the tibia tunnel. When the user finds the position or the direction of the tibia tunnel incorrect in the following operations, she or he can return to this point to adjust the placement of the locator.
(5) *Femur tunnel locating*. This is almost the same case as the restoration point mentioned previously.
(6) *Leading acus inserting*. At this point, a leading acus along with a piece of leading thread needs to be inserted into the knee joint through the arthroscopic portal for instruments and through the femur tunnel to get out of the knee joint. When pulling the leading acus outside the femur tunnel, the user has to make sure that the end of the leading thread is left in the

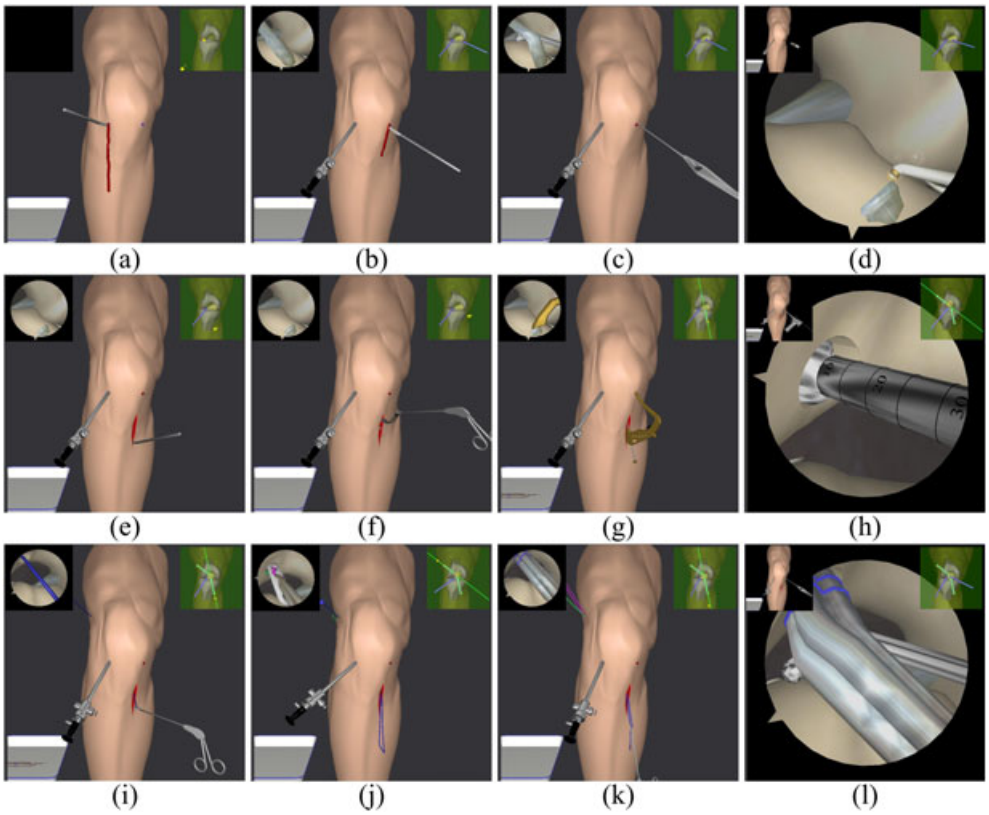


Figure 16. Snapshots of our system taken in (a,b) puncture stage, (c,d) inspection and clean-up stage, (e,f) graft generation stage, (g,h) tunnel construction stage, (i,j) graft placing stage, and (k,l) accomplishment stage.

DOI: 10.1002/cav

knee joint. If this is not the case, it is necessary to undo the operation and try it once again.

7. RESULTS

By integrating the aforementioned methods, we developed an advanced VR-based knee arthroscopy training system. The main goal of vKASS is to provide a simulator for complex surgical procedure. Thus, the most important issues of concern include real-time performance, stability, and the completeness of the surgical procedure being simulated, whereas some other requirements such as accurate physical properties of the tissues can be relaxed. On our system, the training of arthroscopic ACL reconstruction using the EndoButton method can be conducted. Figure 16 shows some screen snapshots during the execution of our system. Compared with surgery scenes in real world, our system provide a realistic environment, which increases the immersive feeling of the user. Moreover, all the crucial operations in the real arthroscopic ACL reconstruction are simulated in our system. Because we keep in mind the efficiency during the development of our system, the real-time performance of our system is more than sufficient for user interaction purpose. Even though there are many issues to be simulated, our system can be executed at a frame rate of more than 30 Hz on the hardware platform mentioned in Section 2.1.

We have tested our system with a small group of surgeons, including 22 novices and 8 experts in arthroscopy. After training, these subjects were asked to evaluate vKASS according to two aspects, "reality" and "usefulness", respectively. Figure 17 shows the evaluation results from these subjects, where a score 0 means totally negative and a score 5 means totally positive. From the results, we can see that all the subjects give positive evaluation to the reality and usefulness of our system. They pointed out that our system provides realistic visual effects and force feedback. We also discovered that trainees who have very limited experience in the real arthroscopic surgery are especially interested in vKASS. They regard vKASS as a good supplement to traditional training methods. On the other hand, experts in arthroscopy find vKASS to be very useful in explaining the overall procedure to novices in a step-by-step manner. They emphasize that vKASS is more efficient and intuitive than existing educational techniques such as video record and multimedia.

8. CONCLUSION

We have developed vKASS, an advanced VR-based knee arthroscopy training system, which can provide training for not only basic surgery skills but also the complex ACL reconstruction procedure. The hardware and software architectures, as well as some key techniques, we used in the development of our system are presented in this paper. The surgical procedure simulated in our system is the standard method for ACL reconstruction, and the simulation results have been justified by experts in arthroscopy. Users of vKASS considered it effective in teaching the theory of the complex surgery and in training for the surgical procedure. However, there still remain several limitations in our system. First, the haptic devices in use can provide only three DoFs of force feedback. Thus, the torque exerted on the instrument cannot be simulated and presented to the user. Second, all the anatomic models are reconstructed from the Visible Human Project data set and cannot be adjusted for special cases. At last, although we got positive feedbacks from users of our system, a quantitative comparison between our VR-based training and classical training methods is missing.

These limitations naturally imply several research directions in the future. We will consider increasing the DoF of the haptic devices by either using other more advanced devices or performing modification on the current ones. *In vivo* computed tomography and magnetic resonance imaging data are considered to be incorporated into our system, so that simulation on specific patient cases can be carried out. Then, our system can serve as a preoperative surgery planning or rehearsal tool to reduce the risk in the real operating theater. More importantly, we will conduct a comparative user study to quantitatively validate the effectiveness of our system.

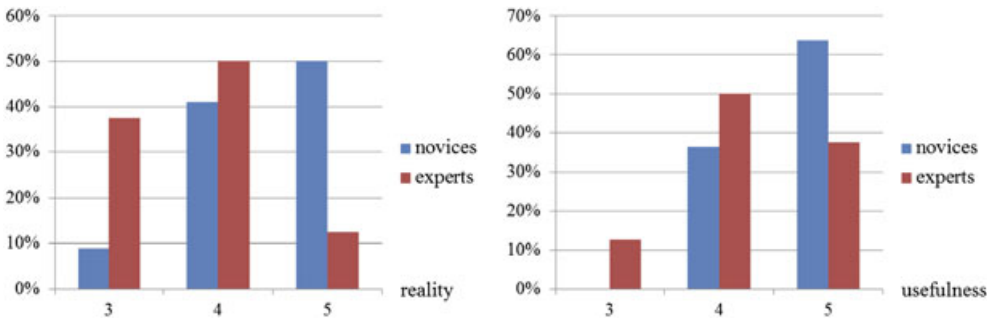


Figure 17. Evaluation results.

ACKNOWLEDGEMENT

The research work in this paper was partially sponsored by the Research Fund for the Doctoral Program of Higher Education with grant number 20104307110003 and the National Science Foundation of China with grant number 60773022.

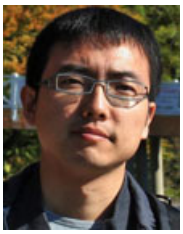
REFERENCES

1. Gibson, S, Samosky, J, Mor, A, Fyock, C, Grimson, E, Kanade, T, Kikinis, R, Lauer, H, McKenzie, N, Nakajima, S, Ohkami, H, Osborne, R, Sawada, A. Simulating arthroscopic knee surgery using volumetric object representations, real-time volume rendering and haptic feedback. In *Proceedings of the First Joint Conference on Computer Vision, Virtual Reality and Robotics in Medicine and Medial Robotics and Computer-Assisted Surgery,* CVRMed-MRCAS '97. Springer-Verlag, London, UK, 1997; 369–378.
2. McCarthy AD, Hollands RJ. A commercially viable virtual reality knee arthroscopy training system. *Studies in Health Technology and Informatics* 1998; **50**: 302–309.
3. Müller W, Bockholt U. The virtual reality arthroscopy training simulator. *Studies in Health Technology and Informatics* 1998; **50**: 13–21.
4. Müller W, Bockholt U, Lahmer A, Voss G, Börner, M. Vrats–virtual reality arthroscopy training simulator. *Radiologe* 2000; **40**(3): 290–293.
5. Megali G, Tonet O, Mazzoni M, Dario P, Vascellari, A, Marcacci M. A new tool for surgical training in knee arthroscopy. In *Proceedings of the 5th International Conference on Medical Image Computing and Computer-assisted Intervention-Part II,* MICCAI '02. Springer-Verlag, London, UK, 2002; 170–177.
6. Mabrey JD, Gillogly SD, Kasser JR, Sweeney HJ, Zarins B, Mevis H, Garret WE, Poss R, Cannon WD. Virtual reality simulation of arthroscopy of the knee. *Arthroscopy: The Journal of Arthroscopic and Related Surgery* 2002; **18**(3): E28.
7. Massie TH, Salisbury KJ. PHANToM haptic interface: a device for probing virtual objects, In volume 55-1 of *Proceedings of the 1994 International Mechanical Engineering Congress and Exposition*, Massachusetts Inst of Technology, Cambridge, United States, ASME, 1994; 295–299.
8. Heng P-A, Cheng C-Y, Wong T-T, Xu Y, Chui Y-P, Chan K-M, Tso S-K. A virtual-reality training system for knee arthroscopic surgery. *IEEE Transactions on Information Technology in Biomedicine* 2004; **8**(2): 217 –227.
9. Funk L, Umaar R, Awan A, Gandi M. Computer simulation training for arthroscopic surgery. *Tameside Medical Journal* 2007; **1**(3): 5–8.
10. Bayona S, Fernández-Arroyo JM, Bayona P, Pastor, L. A new assessment methodology for virtual reality surgical simulators. *Computer Animation and Virtual Worlds* January 2009; **20**: 39–52.
11. GMV. Virtual reality arthroscopy trainer - short description of the product, 2009.
12. Bro-Nielsen M. Finite element modeling in surgery simulation. *Proceedings of the IEEE* March 1998; **86**(3): 490–503.
13. GHOST SDK version 3: programmer's guide, 1998.
14. Ackerman AJ. The Visible Human Project. *Proceedings of the IEEE* 1998; **86**(3): 504–511.
15. Kass M, Witkin A, Terzopoulos D. Snakes: active contour models. *International Journal of Computer Vision* 1988; **1**(4): 321–331.
16. Geiger B. Nuages: a package for 3D reconstruction, 1996.
17. Taubin G. A signal processing approach to fair surface design. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques,* SIGGRAPH '95. ACM, New York, NY, USA, 1995; 351–358.
18. Xu K, Cheng Z-Q, Wang Y, Xiong Y, Zhang H. Quality encoding for tetrahedral mesh optimization. *Computers and Graphics* 2009; **33**(3): 250–261. IEEE International Conference on Shape Modelling and Applications 2009.
19. Müller M, Heidelberger B, Hennix M, Ratcliff J. Position based dynamics. *Journal of Visual Communication and Image Representation* April 2007; **18**: 109–118.
20. Nevatia RN, Binford TO. Structured description of complex objects, In *Proceedings of the Third International Joint Conference on Artificial Intelligence*, Stanford, California, US, 1973; 641–647.
21. Adams B, Dutré P. Interactive Boolean operations on surfel-bounded solids. *ACM Transactions on Graphics* July 2003; **22**: 651–656.

AUTHORS' BIOGRAPHIES

Yanzhen Wang received his PhD degree in Computer Science from the National University of Defense Technology, China, in 2011. He is currently an assistant professor at the College of Information Systems and Management, National University of Defense Technology, China. His research interests include virtual reality, physically based modeling, geometric modeling, and computer animation.

Yueshan Xiong is currently a professor at the School of Computer Science, National University of Defense Technology, China. His current research interests include computer graphics, virtual reality, geometry processing, and numerical computation.



Kai Xu received his PhD degree in Computer Science from the National University of Defense Technology, China, in 2011. He is currently a postdoctoral researcher at Shenzhen Institute of Advanced Technology and an assistant professor at the National University of Defense Technology. His research interests include high-level shape analysis and processing, texture synthesis, and shape deformation.

Dong Liu received his MSc degree in Computer Science from the National University of Defense Technology, China, in 2009. His research interests include virtual reality, geometry processing, and image processing.