

Crack-Free Multi-Resolution Dynamic Volume Refinement for Physics-Based Bone Drilling and Shaving in Arthroscopic Procedures

Mustafa Tunc*
Google

Doga Demirel†
School of Computer Science,
University of Oklahoma

Sinan Kockara‡
Department of Computer Science,
Rice University

Tansel Halic§
Intuitive Surgical Inc.

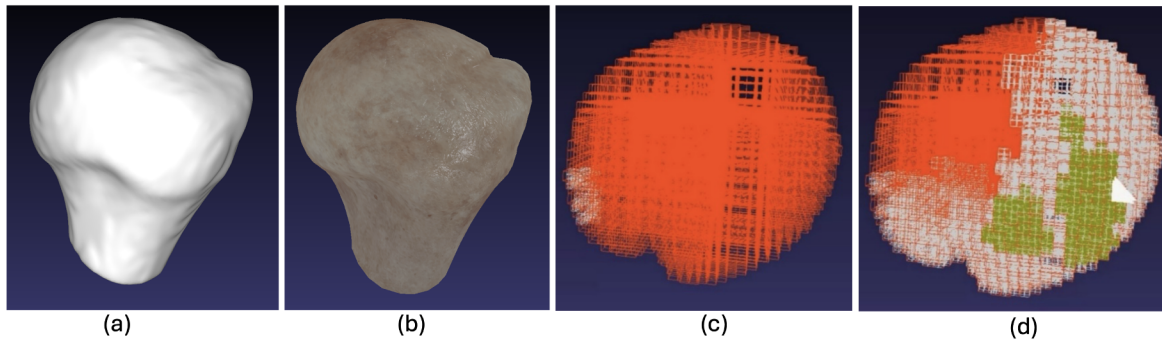


Figure 1: (a) Surface mesh of the humeral head generated using the Marching Cubes algorithm, (b) Original 3D model of the humeral head, (c) Sub-voxel creation at level -1 and (d) level -2. The white triangle indicates the haptic interaction point within the scene.

ABSTRACT

Arthroscopy is a minimally invasive procedure used by orthopedic surgeons to diagnose and treat joint injuries. It involves using a 2D fiber optic view from an arthroscope displayed on a monitor, requiring high dexterity and practice. This study has two main goals: to develop a virtual reality training platform called Virtual Rotator Cuff Arthroscopic Skill Trainer (ViRCAST) for diagnosing and treating rotator cuff tears, and to introduce a novel algorithm for simulating bone shaving and drilling in arthroscopic surgery. ViRCAST and its hardware components are described, and realistic haptic interaction is achieved using Dynamic Proximity Hierarchy's multi-point collision detection. A dynamic resolution approach improves haptic force feedback by generating finer voxels based on proximity to the surgical tool. The bone drilling simulation ensures smooth feedback even with coarse-resolution voxels. With a voxel size of 1.2mm, execution time is 6 ms, while reducing the voxel size to 0.54mm results in surface mesh creation in 34 ms and visual rendering at 29 FPS.

Index Terms: Arthroscopic Rotator Cuff, Voxels, Arthroscopy, Surgery Simulator, Dynamic Voxelization, Real-time Bone Drilling Simulation.

1 INTRODUCTION

Arthroscopy is a minimally invasive surgical operation where orthopedic surgeons assess, diagnose, and treat injuries inside of a joint space [29]. Arthroscopic rotator cuff repair is a surgical procedure targeting the four muscles—supraspinatus, infraspinatus, subscapularis, and teres minor—that link the upper arm to the shoulder

blade [12]. During the procedure, surgeons use pencil-sized instruments equipped with a small lens and light source, inserting them into the joint to visualize the anatomy on a 2D monitor. The live feed is provided by an arthroscope, a rigid fiber optic camera with an integrated light source. Training for arthroscopic techniques is particularly challenging due to the non-intuitive hand-eye coordination, limited tool control, and a limited field of view [27]. Conventional models such as cadavers, apprenticeship model, and mannequins are used for training [35, 2], but they are neither effective nor adequate. The most common model, the apprenticeship model where the novice surgeons carry out some parts of the surgery [35]. This model bears risks for patients and has zero tolerance for mistakes. Physical bench models are valuable for practicing arthroscope and instrument navigation, however, they lack in providing anatomical realism necessary for teaching joint anatomy or assisting in surgical decision-making [28]. Compared to the traditional training methods, surgical simulators—whether virtual reality (VR) or physical platforms—provide realistic, cost-effective, and risk-free training environments, making them a superior alternative to traditional training methods.

VR-based medical simulations have gained popularity in medical education with the emergence of VR and haptic feedback devices [31, 32]. VR training platforms provide surgeons with an affordable and realistic way to enhance their manual dexterity [4, 6]. Additionally, these platforms enable user performance assessment without requiring input from expert surgeons. VR-based teaching methods are equally effective as direct observation, cadaver models, and video-based learning tools [14, 17, 26]. In standard arthroscopy, surgeons must master essential skills such as camera navigation (camera dexterity and field of view) [10, 11], bimanual instrument control, scope adjustment, and object manipulation [15, 22, 5]. Additionally, surgery-specific tasks for arthroscopic rotator cuff repair like bone grafting, drilling anchor holes, and inserting anchors into bone are critical components of the learning curve [1]. VR simulators offer an effective platform for both instruction and evaluation [31], with the added benefit of providing haptic feedback during interactions between instruments and anatomy. This allows for realistic and precise tactile sensations

*e-mail: mustafatunc@google.com

†e-mail: doga@ou.edu

‡e-mail: skockara@rice.edu

§e-mail: tansel.halic@intusurg.com

when performing tasks within the joint [26]. Bone shaving, burring, and drilling are all essential tasks in arthroscopy, each requiring dynamic surface manipulation and real-time adjustments to 3D geometry. Generating a dynamic surface is a computationally intensive process that demands considerable computational resources. Integrating haptic force feedback during contact, such as bone contact with a shaver, further amplifies the complexity of this task. This complexity arises from the necessity to deliver robust forces at acceptable execution rates, for instance, meeting the requirement of a force computation frequency of 1 KHz, during real-time and dynamic surface modifications occurring when the surgical instrument interacts with the bone (e.g., during bone drilling, shaving, or burring) within a VR scene. This multifaceted problem constitutes the most challenging issue in VR-based surgery simulations and acts as a constraint for real-time performance. In a virtual environment, volumetric models such as bone or tissue are represented by voxels occupying cubic volumes. Voxels serve as 3D representations of pixels in 3D space and are preferred for encapsulating volumetric spaces as they capture both internal and external structures, unlike surface models that only depict outer surfaces without internal content.

The objective of this study is to devise an efficient algorithm capable of dynamically updating surfaces, managing collisions, and calculating robust force feedback in real-time. Tactile feedback essential for perceiving touch and drilling sensations is computed based on the interactions between voxels and the surgical tool. Achieving precise 3-D representations and force feedback necessitates small voxel sizes (*VS*), thereby imposing significant computational demands on the system. To address this challenge, we propose a novel method that dynamically generates small voxels based on the instrument's distance to the bone. The proposed approach has been implemented and tested within the Virtual Rotator Cuff Arthroscopic Skill Trainer (ViRCAST). [6, 11] simulator.

2 BONE INTERACTION SIMULATIONS

The majority of literature on bone shaving, burring, and drilling simulations in VR originates from dental procedure simulators. The prevalent approach involves representing shapes with volumetric structures called voxels, which can be removed upon contact, followed by surface reconstruction algorithms such as MC [23] for polygonal surface visualization.

Wang et al. [33] proposed iDental, a prototype system incorporating real-time drilling simulation using a PHANToM haptic device, which could replicate several manual dexterity exercises and provide force feedback. To address instability from vanishing contact during drilling, they devised a method that removes overlapping voxels and redefines boundary voxels, enabling accurate force feedback. Wu et al. [34] proposed a drilling simulation using 0.1 mm voxels and CT images for multi-material tooth modeling, enhancing realism in dental surgery. The collision detection algorithm used, similar to Wang et al.'s [33] method, used boundary voxels for force feedback and interior voxels for material removal. It calculated tactile feedback based on the drill's movement and transmitted the cumulative forces to the haptic device. Morris et al. [24] developed a temporal bone dissection platform using 3D models generated from patient-specific CT data. A 6-DOF haptic feedback system guided the removal of drilled voxel regions and redundant triangles. Petersik et al. [25] conducted a petrous bone surgery simulation using multi-point collision detection and CT-scan images to generate a 3D model, with point clouds on the drill's surface guiding collision determination and force feedback calculation.

Achieving a balance between precise force feedback and acceptable visual rendering remains to be a major challenge. Existing methods utilize static *VS*, which struggle to maintain the precision of tactile sensations alongside real-time performance, as small *VS*—required for accuracy—substantially increase computa-

tional demands. Additionally, all current methodologies [33, 34, 24, 25, 30, 3] rely on the MC algorithm [23] for isosurface extraction. MC is the de facto standard for isosurface extraction in voxel-based methods, but it has inherent limitations. A notable issue is the "hole/crack problem," which arises from approximating isosurfaces with polygonal meshes [9]. This can result in cracks, particularly along curved surfaces, due to inconsistent vertex interpolation across adjacent cubes. These cracks lead to missed collisions and inconsistent force feedback, undermining the realism and plausibility of the simulation. To the best of our knowledge, previous works have not effectively addressed this issue. To resolve both the fixed *VS* and the crack problem, we propose a dynamic voxelization algorithm that adjusts voxel resolution based on the proximity of the medical instrument to the bone. This approach allows for efficient real-time updates while ensuring smooth and continuous force feedback, representing a significant improvement over existing methods. The dynamic proximity hierarchy (DPH) adaptively generates smaller voxels near the drill, thereby significantly reducing mesh cracks and enhancing the precision of force feedback, while simultaneously rendering bone surfaces with smooth, high-resolution triangle meshes.

3 METHODS

3.1 ViRCAST

The ViRCAST hardware interface replicates a patient's shoulder and the setup of an operating room for shoulder arthroscopy procedures. It includes a frame structure resembling an operating table, a mannequin arm with surgical portals, a monitor for visualizing the shoulder's operational area, and authentic surgical instruments, enhancing realism for trainees. Fig. 2 depicts the integrated hardware component of the platform.

In the simulator, trainees can navigate a shoulder model using a real arthroscope connected to a Geomagic Touch haptic device (see Fig. 3a). ViRCAST uses a modified arthroscope with two rotational encoders for independent control over the arthroscope rotation and the light source, enabling seamless motion detection and translation to the virtual environment. The interface also supports attaching a probe to the haptic device and motorized instruments like the shaver and burr, controlled by buttons fixed on the instruments. 3D-printed interfaces connect the instruments to the haptic device and buttons, with each button and encoder connected to an Arduino Uno for autonomous operation. Input values are read at a baud rate of 9,600 and transmitted to the simulator, where they are parsed and executed for functions like camera rotation or shaver activation.

In rotator cuff repair surgery, the initial step is to locate and assess the tear, which ViRCAST simulates using a 3D supraspinatus model with a crescent-shaped tear. We used Nvidia's PhysX to simulate the soft tissue structure of the tear, allowing trainees to probe, manipulate, and assess the tear. The tear exhibits elastic behavior but is laterally constrained to prevent movement away from the midline, ensuring realistic deformation while remaining connected to the humeral head.

Ensuring visual parity between actual surgery and simulation is crucial, achieved using physically based rendering (PBR) techniques. A spotlight affixed to the arthroscope illuminates the scene, and PBR calculations, based on the spotlight's parameters and the bidirectional reflective distribution function (BRDF), account for light directions, surface normals, and microfacet roughness, with textures containing surface parameters. As surface parameters influence the appearance of objects at specific points on a 3D model, we incorporate textures containing surface parameters for each surface pixel into the PBR pipeline.

In arthroscopy, procedures such as shaving and drilling can lead to bleeding, necessitating to control the bleeding through adjustments in fluid pressure or electrocautery. ViRCAST utilizes smoothed particle hydrodynamics (SPH) to simulate continuous

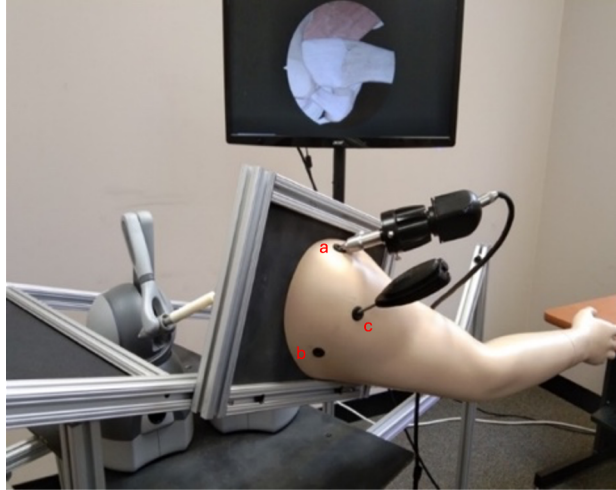


Figure 2: ViRCAST Simulator - showing the simulator's view, shoulder mannequin, surgical portals (a. Anteroinferior (5 o'clock), b. Posteroinferior (7 o'clock), and c. Lateral Subacromial), and haptic devices attached to surgical instruments.

fluid flow and heat transfer within a single framework, crucial for maintaining a clear camera view and cooling the surgical volume during electrocautery in real-time [8]. This helps to accurately simulate the fluid motion of the irrigation solution and the associated heat transfer mechanisms of conduction and convection within the shoulder cavity.

Simulating bone deformation during the drilling of a suture anchor into the humeral head in arthroscopic rotator cuff surgery poses several computational challenges in VR, such as regenerating the 3D mesh, providing force feedback, detecting collisions, and ensuring real-time rendering. Volumetric 3D models using voxels are widely employed due to their ability to represent volume changes with advantages in surface construction, haptic interaction, and memory efficiency, albeit requiring significant computational resources for accurate geometric representation with small VS . To address the challenges associated with existing voxel-based approaches for simulating rigid and deformable body interactions requiring frequent volume changes such as dissection or drilling, we propose a novel method utilizing DPH [21, 20] to enhance voxel-based simulations of rigid and deformable body interactions. DPH allows for dynamic adjustment of VS based on interactions with the drill object, ensuring accurate haptic force feedback calculation and efficient simulation of anchor placement tasks in ViRCAST.

3.2 Voxelization

A voxel functions as a three-dimensional counterpart to a pixel, crucial for generating detailed interactions and enabling accurate force feedback in volumetric environments, unlike surface models which lack internal content. Each voxel contains data such as its center, size, and additional properties like bone density in surgical simulations or color values in gaming environments. By focusing computational resources on regions relevant to drilling and shaving procedures within the humeral head model, we aim to significantly improve simulation performance. This voxelization process begins with establishing axis-aligned bounding boxes and placing voxels based on ray projections, ensuring efficient balance between simulation accuracy and visual rendering capabilities through careful VS selection.

3.3 Dynamic Proximity Hierarchy

Following voxelization, organizing voxels into three-dimensional arrays is essential, yet for efficient simulation, a robust hierarchy like DPH [21, 20, 7] is crucial for rapid calculations at runtime. In ViRCAST, we employ DPH for collision detection and multi-resolution haptic rendering, utilizing its spanner tree hierarchy to manage voxel interactions within our volumetric models. Originally designed for soft bodies and deformations, DPH dynamically updates to accommodate changes in mesh structures, though our bone drilling simulation focuses on rigid bodies without elastic deformation. Instead, we adapt DPH's dynamic proximity tracking to optimize voxel-based scenarios, modifying nodes to enhance simulation accuracy while maintaining DPH's fundamental properties [21].

Voxels generated based on the features outlined in [21, 20] serve as the base nodes ($level 0$) for DPH construction. Fig. 3 illustrate the hierarchical process using a 2D example with squares representing voxels, and an expansion ratio $\xi = 3$, five nodes ($V_0 = 7, 8, V_1 = 8, 8, V_2 = 9, 8, V_3 = 9, 7, V_4 = 10, 6$) are positioned at $level 0$, shown as red squares in Fig. 3. During DPH construction, the $level 0$ voxels (V_0) are directly inserted as nodes at the lowest level ($level 0$), which corresponds to the highest resolution. The VS at this level are designated as $\xi^0 = 1$. The locations of the $level 0$ nodes (voxels) are shown by the red squares in Fig. 3.

Hierarchy construction proceeds by building the second level, starting with the selection of a randomly chosen node at $level 0$. It's worth noting that varying the random ordering will result in different yet equally effective hierarchies. Since V_0 does not require an expansion calculation due to the absence of $level 1$ nodes in the hierarchy, a new node, V_{y_0} , is generated. This new node uses the same center as V_0 and occupies $\xi^1 = 3$ cells (depicted by the black square in Fig. 3a). Moving on to V_1 , as its center lies within the parent voxel V_{y_0} , it automatically becomes a child of V_{y_0} and is thus skipped. Subsequently, V_2 is evaluated, finding that it is not yet covered by any black voxel (representing upper-level voxels). Therefore, it expands to the next level by creating V_{y_1} , utilizing the same location as the center of the upper-level voxel (as illustrated in Fig. 3b). V_3 is designated as a child of V_{y_1} due to its center being within it. Lastly, V_4 , not situated within any of the $level 1$ voxels, expands to the upper level, V_{y_2} (see Fig. 3c). With no nodes left at $level 0$, construction of $level 1$ is complete. Next, the assembly of $level 2$ begins. We begin by checking the first node in the $level, 1$ list. V_{y_0} generates its $level 2$ node, V_{b_0} , with a parent node at $VS \xi^2 = 9$, using its center (as shown by the blue square in Fig. 3d). Since the remaining $level 1$ voxels are already covered by V_{b_0} , they are assigned as V_{b_0} 's children. The hierarchy construction concludes with only one node (voxel) at this level, designating it as the root node of the hierarchy. DPH creation exhibits a time complexity of $O(n \log n)$, where n represents the number of voxels.

Bone voxels with colors representing different levels in the DPH structure are depicted in Fig. 4 - red ($level 0$), yellow ($level 1$), green ($level 2$), blue ($level 3$), and purple ($level 4$). The purple voxel (Fig. 4a) serves as the root node, covering the entire model as the sole node at its level.

3.4 Surface Creation

Realistic visualization is crucial for simulation validity, especially in surgical contexts. Although direct volume rendering of voxels is used in certain applications, it often fails to deliver realistic visualization in surgical simulations, even when utilizing very small VS . Transfer functions for shading [18, 19] and real-time surface generation algorithms are essential to avoid blocky voxel rendering. Methods like MC [23], Surface Nets [13], and Dual Contouring [16] are commonly used for surface extraction. Marching Cubes is utilized as the standard isosurface extraction algorithm, ensuring accurate polygonal surface visualization post-interaction in surgical

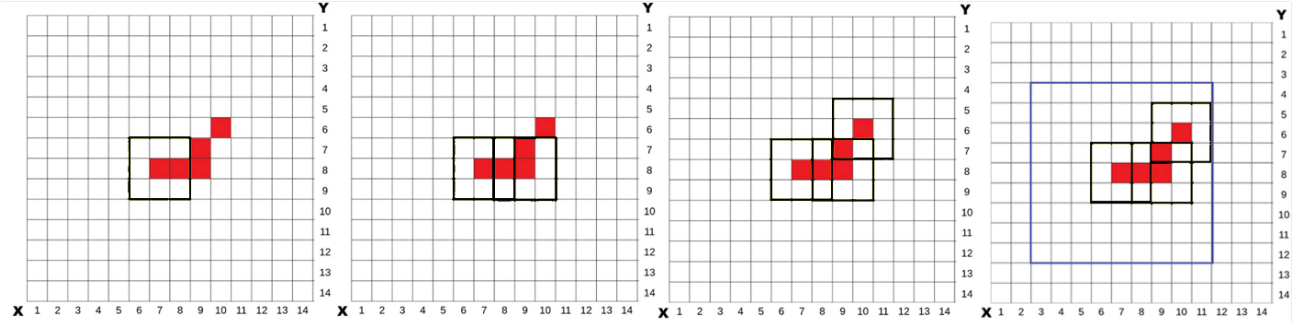


Figure 3: (a) Insertion of *level 1* nodes at {7,8} (b) {9,8}, (c) {10,6} into the hierarchy, (d) insertion of the *level 2* voxel into the hierarchy.

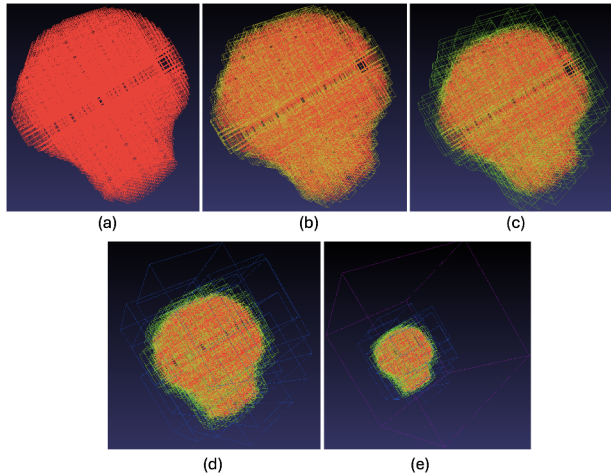


Figure 4: Steps for hierarchy assembly from (a) level 0 to (e) level 4.

scenarios.

MC algorithm relies on precise iso-value distribution to extract robust surfaces by creating triangles within each voxel based on iso-values assigned to its corners. After generating the DPH hierarchy, we assign iso-values to *level 0* voxels and ensure the entire 3D geometry is represented accurately without shrinking at boundaries. During MC execution, triangles are generated voxel by voxel, forming a complete mesh stored with vertex positions for accurate spatial representation. To address square faces and sharp edges, we employ Laplacian smoothing using vertex neighbors stored in a hash table. This smoothing process enhances visual fidelity by eliminating bumps and sharp edges, as depicted in Fig. 1a and Fig. 1b. By forwarding all edge voxels to MC and ensuring vertex indices match, we prevent surface cracks, crucial for realistic force feedback in haptic interactions during surgical simulations. This regeneration process occurs dynamically with changes in the hierarchy, updating voxel status and preserving hash table values for efficiency.

To analyze and quantify the cracks, we implemented a specialized method. The primary approach for counting cracks involves calculating the percentage of cracks visible on the screen. A double-sided shader was utilized to visualize the cracks, producing a white color for triangles facing the camera and a red color for triangles facing away. Then screenshots of the models are taken from all the angles from 6 sides on a black background. White pixels represent the surface of the humerus, while red pixels indicate the interior surface exposed due to cracks on the outer surface. To cal-

culate the visible crack value, the coverage of red pixels on the surface is determined. Black pixels, representing the background, are excluded from the calculation. The remaining counts of white and red pixels are then used to calculate crack coverage over all surface area using the formula: $\frac{\text{pixel}_{\text{red}}}{\text{pixel}_{\text{red}} + \text{pixel}_{\text{white}}} \times 100$. To evaluate crack coverage on both surfaces (MC vs. DPH), pixel values from each image were analyzed. The red pixels covered 10.945% of the surface generated by the MC algorithm, compared to only 0.025% for the surface produced by the DPH approach. These results demonstrate that our method significantly reduces surface cracks, yielding approximately 437 times fewer collision misses compared to the widely used MC algorithm. This reduction in collision misses greatly enhances the realism of real-time simulations. Cracks or holes in the surface lead to missed collisions, which in turn prevent the generation of force feedback. When a collision is finally detected in crack-free regions, it causes a sudden force feedback jump, leading to abrupt force fluctuations. Additionally, a higher number of surface cracks can result in erratic and jagged haptic force feedback. Therefore, minimizing surface cracks leads to smoother and more consistent force feedback, improving the overall plausibility of the simulation.

3.5 Collision Detection and Haptic Force Feedback

Ensuring realistic haptic rendering requires precise collision detection between the bone and surgical tools, crucial for accurate force transmission at haptic rates (around 1 kHz) [22], as detailed in Fig. 1c and Fig. 1d. Optimal VS are essential for achieving accurate bone drilling simulations, balancing resolution with real-time performance constraints. While smaller voxels enhance accuracy, their real-time implementation challenges are addressed through dynamic resolution adjustments based on surgical instrument proximity. The octree algorithm is employed alongside DPH to dynamically adjust voxel resolution beyond *level 0*, ensuring both computational efficiency and precise force feedback calculation, as illustrated in Fig. 5a.

The trade-off between VS impacts both performance, graphics quality, and force calculation accuracy. Larger voxels enhance computational efficiency but may compromise precision. Default regular VS are used with dynamic adjustments for optimal resolution. DPH doesn't support additional hierarchy at *level 0*, necessitating the octree algorithm to dynamically adjust voxel resolution deeper than *level 0* in real-time (Fig. 5a). Base nodes are divided into smaller voxels up to two additional levels based on proximity to the surgical instrument, generating up to 64 octree nodes. The octree depth is defined by $D = \max(0, L - 2)$ where L is the node level in DPH. Fig. 1c and Fig. 1d illustrates this process with white (*levels - 1*), green *levels - 2*, and pink *levels - 3* voxels. Octree nodes remain active as long as the instrument is near *level 0* voxels; nodes are deleted after a designated timeout period to manage mem-

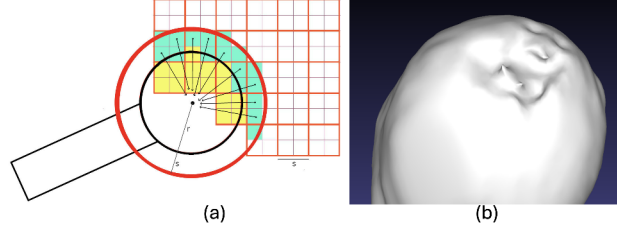


Figure 5: (a) Depiction of the drilling tool geometry, (b) surface of the humeral head post-drilling.

ory efficiently, based on spatial and temporal coherence (Fig. 1c and Fig. 1d).

We adopted a modified multi-point method for virtual drill tools inspired by Wu et al. [34], replacing points with voxelized models of both the tool and tooth. Collision detection focuses on edge voxels of the tool and tooth model voxels, triggering force calculation and material removal, respectively. This approach ensures continuous force feedback across the entire tool surface, enhancing computational efficiency with a two-layer sphere model for the drill instrument. The inner layer handles material removal within the tool's spherical volume, while the outer layer acts as a cushion, minimizing force discontinuities (Fig. 5a). Grid-shaped bone voxels are depicted in orange (*level* 0) and purple (*level* -1), with black lines that outline the geometry of the drill tool and a red circle representing the outer layer of the tool's tip. Cyan voxels between the layers are used for force calculation, with arrows indicating their normal directions towards the tool's center.

The direction of the force was determined by summing the direction vectors from the centers of the voxels within the collision space between the two spheres. The direction vector is defined as:

$$\vec{F}_{\text{direction}} = \frac{\sum_{k=1}^n (C_{\text{Tool}} - C_{\text{Voxel}_k})}{\left| \sum_{k=1}^n (C_{\text{Tool}} - C_{\text{Voxel}_k}) \right|} \quad (1)$$

where C_{Tool} is the center of the tool, C_{Voxel_k} and n are the center of the k th voxel and the number of the sub-zero level voxels between the two layers. The force magnitude was determined by estimating the maximum number of voxels that fit between the layers and counting those within the collision space in each simulation frame. This estimation is precomputed as follows:

$$M = (N_{\text{max}}) \frac{\frac{4\pi r_o^3}{3} - \frac{4\pi r_i^3}{3}}{\frac{4\pi v_{\text{min}}^3}{3}}, \quad c = (N_{\text{max}}) \frac{r_o^3 - r_i^3}{v_{\text{min}}^3} \quad (2)$$

where r_o is the radius of the outer sphere, r_i is the radius of the inner sphere, v_{min} is the minimum VS at below zero levels, c is a material-specific constant, and N_{max} is the maximum force the haptic device can exert (3.3N). These calculations result in the final force measure transmitted to the haptic device, which is calculated as: $\vec{F} = M\vec{F}_{\text{direction}}$. We introduced a unitless density attribute d for bone voxels to indicate drilling difficulty, initialized as an integer. Each drilling operation reduces this value ($d - 1$), and when a voxel's $d \leq 0$, it is removed from the hierarchy. Fig. 5b illustrates snapshots of the humeral head at different stages of drilling.

4 RESULTS

For each data recording trial, we carried out two runs of the drilling simulation tests: first with our dynamic resolution algorithm enabled, using various VS, and then with the algorithm disabled. The drilling path was procedurally defined, starting from two surgical portal locations: Anteroinferior (5 o'clock) and Posteroinferior (7 o'clock), as shown in Fig. 2. The instrument followed these paths

while drilling into the bone at a 45-degree angle. Moving from the outer edge of the humeral head to the most distant point on the path, the virtual drill moved at a constant speed, with each path taking seven seconds to trace and record force data. The force data was logged for analysis. Throughout the sessions, the inner sphere of the drill maintained a constant size, with a diameter of $d = 2.92\text{mm}$.

The initially generated volumetric bone model had a low resolution of (VS) = 1.2mm. In the case of dynamic-resolution, we applied a 2-level division of the base voxels. When the tool moved near the bone, the closest voxels increased in resolution by forming an octree with a depth of 2. As a result, although we initially had a VS of 1.2mm, the actual size used for force calculation was $\frac{1.2\text{mm}}{2^2} = 0.3\text{mm}$. Another experiment performed with the 1-level division yielded $\frac{1.2\text{mm}}{2^1} = 0.6\text{mm}$ sub-VS. Fig. 6 shows the force plots for static and dynamic VS with 1- and 2-level divisions at two different portals, respectively.

After the experiment with 1.2mm voxels, we reduced the VS to 0.9mm. When we increased the voxel resolution, we determined that generating more than 1 level below zero level voxels was necessary as VS picked was small enough to provide smooth force feedback, which was $\frac{0.9\text{mm}}{2^1} = 0.45\text{mm}$. Fig. 7 demonstrate the force plots for the 0.9mm VS in both drilling directions.

The final experiment was for the VS 0.72mm with doubled resolution, resulting in a VS of $\frac{0.72\text{mm}}{2^1} = 0.36\text{mm}$. In the simulation, we utilized three separate threads for collision detection, and visual and haptic rendering. For all experiments, the haptic and collision detections threads were updated at a rate of 1KHz to avoid unnecessary computations and ensure synchronization for consistent force feedback. Each time a voxel was removed from the hierarchy, the triangle index buffer was recalculated which slowed down the visual rendering thread. Mesh extraction time was influenced by both the VS and the amount of voxels. Tab. 1 presents the VS, average mesh regeneration time and average FPS for graphics rendering.

4.1 Discussion

As shown in Fig. 6 through Fig. 8, we obtained consistent drilling force values using static and adaptive voxels. The graphs demonstrate the increases and decreases in force magnitude resulted within comparable time periods. This indicates that the drill positions were accurately aligned within the bone during the recording of the force data. Upon examining the force plots (Fig. 6 - Fig. 8) for fixed-size voxels, it is evident that larger VS led to more significant fluctuations in force compared to smaller VS. From Fig. 6 to Fig. 8, the change in force values between the 2nd and 5th seconds was approximately 0.5N, 0.3N, and 0.1N, respectively. As a result, smaller VS produced smoother force feedback than larger ones.

Tab. 1 presents the initial number of voxels, triangles, and vertices extracted when the simulation began. Mesh regeneration involved several subroutines, such as smoothing and updating vertex normals, each with varying execution times. The details of these mesh generation sub-tasks and their respective execution times are shown in Tab. 2.

Compared to the static method, dynamic method generating smaller voxels for force computation resulted in reduced jitter and sudden jumps. Fig. 6b and Fig. 7b illustrate the differences between the two approaches. The adaptive voxel method provided smoother and more stable force feedback as the VS decreased. When we reduced the VS from 1.2,mm to 0.72,mm, the sudden jumps and fluctuations in force almost disappeared. Additionally, generating new voxels with $level < -1$ (creating an Octree with $depth \leq 2$) resulted in even smoother force feedback, even though the initial larger VS. Fig. 6 shows the smoother force feedback produced at $level - 2$ compared to $level - 1$ voxels.

VS didn't have an affect on the force calculation (collision detection) and haptic threads as they remained around 1KHz at all

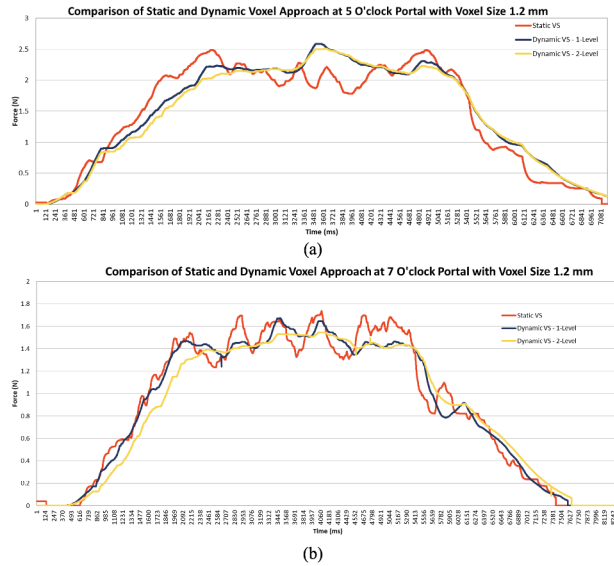


Figure 6: Static and dynamic voxel approach comparison for $VS = 1.2mm$: (a) at 5 o'clock portal and (b) at 7 o'clock portal.

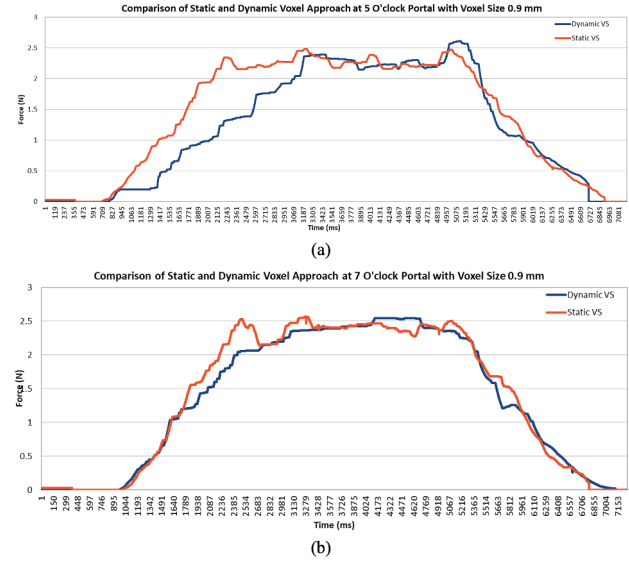


Figure 7: Static and dynamic voxel approach comparison for $VS = 0.9mm$: (a) at 5 o'clock portal and (b) at 7 o'clock portal.

times. The main bottleneck was updating the triangle index buffer and mesh smoothing. For instance, when testing with $VS = 1.2mm$, the mesh replacement required $6ms$, where as $VS = 0.54mm$ required $34ms$. This reduced the FPS for visual rendering from $\lfloor \frac{1000ms}{6ms} \rfloor = 166$ to $\lfloor \frac{1000ms}{34ms} \rfloor = 29$. For smooth rendering, the graphics update rate needs to be at least $24Hz$. Regardless of dynamic or static force computation method, the smallest VS that could initially be used was $0.54mm$. Mesh construction used base voxels ($level = 0$) in both methods, so applying dynamic or static resolution did not affect mesh creation.

In conclusion, the force plots in Fig. 6 -Fig. 8 along with the results in Tab. 1 - Tab. 2 showed that: 1)As expected, smaller voxels provide smoother force feedback than larger voxels, regardless of dynamic or static force computation method, 2) For force computation, dynamically generated voxels produce smoother forces (making the simulation more realistic) compared to static-size voxels, 3)Generating voxels with $level = -2$ in the Octree resulted in even smoother forces with the dynamic resolution method, 4)The smallest VS that could be used for acceptable visual rendering was $0.54mm$, 5) Triangle mesh generation time was consistent for a given VS , irrespective of the force computation method used..

Table 1: Comparison of VS size for mesh generation times, FPS, and the number of voxels, triangles, and vertices.

VS (mm)	Mesh Generation Time(ms)	FPS	# of Voxels	# of Triangles Extracted	# of Vertices Extracted
1.2	6	166	1674	1732	868
0.9	14	71	4569	3736	1870
0.72	20	50	7747	5400	2702
0.54	34	29	15457	9212	4603

Table 2: VS vs. mesh generation execution times.

Execution Step	$VS=0.54mm$	$VS=0.72mm$	$VS=0.9mm$	$VS=1.2mm$
MC and Triangle Creation	15ms	9ms	6ms	2ms
Mesh Smoothing with Laplacian Algorithm (2 iterations)	13ms	8ms	5ms	2ms
Updating Triangle and Vertex Normals	2ms	1ms	1ms	1ms
Clearing Hash Table Values	4ms	2ms	2ms	1ms
Total Execution Time	34ms	20ms	14ms	6ms

5 CONCLUSION

In this study, a virtual arthroscopic rotator cuff surgery simulator (ViRCAST) and its integrated hardware components are presented. Additionally, a virtual bone drilling simulation for arthroscopic rotator cuff surgery is introduced, facilitated by a multiresolution approach for haptic rendering. The humerus bone structure was created using a soft kinetic data structure called Dynamic Proximity Hierarchy (DPH). By utilizing DPH, a multiresolution marching cubes algorithm with various VS at different hierarchical levels was generated, enhancing both broad and narrow phase collision detection.

The proposed approach ensures smooth force feedback even with coarse voxel representation of the bone geometry. Furthermore, for the drilling simulation, visual rendering of the humeral head was improved by applying the multiresolution model for surface generation, alongside multi-pass Laplacian smoothing for better geometric fitting. Integrating negative levels of DPH with an Octree representation enabled the efficient generation of higher resolutions (smaller voxels), thereby achieving real-time performance and robust, high-fidelity force feedback for bone drilling and shaving tasks in virtual reality environments.

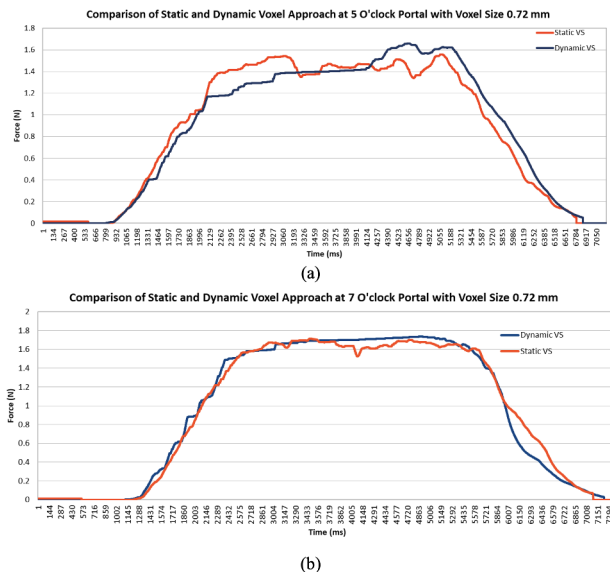


Figure 8: Static and dynamic voxel approach comparison for $V/S = 0.72\text{mm}$: (a) at 5 o'clock portal and (b) at 7 o'clock portal.

ACKNOWLEDGMENTS

This project is supported by the National Institutes of Health (NIH)/NIAMS R44AR075481, NIH/NIBIB R01 EB033674, and R01 EB032820.

REFERENCES

- [1] R. L. Angelo, R. K. Ryu, R. A. Pedowitz, W. Beach, J. Burns, J. Dodds, L. Field, M. Getelman, R. Hobgood, and L. McIntyre. A proficiency-based progression training curriculum coupled with a model simulator results in the acquisition of a superior arthroscopic bankart skill set. 31(10):1854–1871. 1
- [2] V. L. Barker. CathSim. 62:36–37. 1
- [3] S. Delorme, D. Laroche, R. DiRaddo, and R. F. Del Maestro. NeuroTouch: a physics-based virtual simulator for cranial microneurosurgery training. 71:ons32–ons42. Publisher: LWW. 2
- [4] D. Demirel, K. L. Butler, T. Halic, G. Sankaranarayanan, D. Spindler, C. Cao, E. Petrusa, M. Molina, D. B. Jones, and S. De. A hierarchical task analysis of cricothyroidotomy procedure for a virtual airway skills trainer simulator. 212(3):475–484. 1
- [5] D. Demirel, B. Palmer, G. Sundberg, B. Karaman, T. Halic, S. Kockara, N. Kockara, M. E. Rogers, and S. Ahmadi. Scoring metrics for assessing skills in arthroscopic rotator cuff repair: performance comparison study of novice and expert surgeons. 17(10):1823–1835. doi: 10.1007/s11548-022-02683-3 1
- [6] D. Demirel, A. Yu, S. Cooper-Baer, A. Dendukuri, T. Halic, S. Kockara, N. Kockara, and S. Ahmadi. A hierarchical task analysis of shoulder arthroscopy for a virtual arthroscopic tear diagnosis and evaluation platform (VATDEP). 13(3):e1799. doi: 10.1002/rct.1799 1, 2
- [7] D. Demirel, A. Yu, T. Halic, and S. Kockara. Web based camera navigation for virtual pancreatic cancer surgery: Whipple surgery simulator (VPanSS). In 2014 IEEE Innovations in Technology Conference, pp. 1–8. doi: 10.1109/InnoTek.2014.6877375 3
- [8] A. Dendukuri, M. Tunc, D. Demirel, S. Kockara, and T. Halic. Unified framework for real-time fluid simulation in virtual rotator cuff arthroscopic skill trainer (ViRCAS). In 2023 IEEE 23rd International Conference on Bioinformatics and Bioengineering (BIBE), pp. 346–353. IEEE Computer Society. 3
- [9] F. Dinc, K. Oumimoun, W. Kwabla, S. Kockara, T. Halic, S. Arikatla, and S. Ahmadi. Towards real-time bone drilling simulation for anchor

placement in VR based arthroscopic rotator cuff surgery simulation. 2022:178. Publisher: American Medical Informatics Association. 2

- [10] J. Farmer, D. Demirel, R. Erol, D. Ahmadi, T. Halic, S. Kockara, V. S. Arikatla, K. Sexton, and S. Ahmadi. Systematic approach for content and construct validation: Case studies for arthroscopy and laparoscopy. 16(4):e2105. Publisher: Wiley Online Library. 1
- [11] J. Farmer, M. Tunc, D. Ahmadi, D. Demirel, T. Halic, S. Arikatla, S. Kockara, and S. Ahmadi. Virtual rotator cuff arthroscopic skill trainer: Results and analysis of a preliminary subject study. In Proceedings of the 2020 the 4th International Conference on Information System and Data Mining, pp. 139–143. 1, 2
- [12] L. Favard, G. Bacle, and J. Berhouet. Rotator cuff repair. 74:551–557. 1
- [13] S. F. F. Gibson. Constrained elastic surface nets: Generating smooth surfaces from binary segmented data. In W. M. Wells, A. Colchester, and S. Delp, eds., Medical Image Computing and Computer-Assisted Intervention — MICCAI'98, Lecture Notes in Computer Science, pp. 888–898. Springer. doi: 10.1007/BFb0056277 3
- [14] R. F. Henn III, N. Shah, J. J. Warner, and A. H. Gomoll. Shoulder arthroscopy simulator training improves shoulder arthroscopy performance in a cadaveric model. 29(6):982–985. 1
- [15] A. Insel, B. Carofino, R. Leger, R. Arciero, and A. D. Mazzocca. The development of an objective model to assess arthroscopic performance. 91(9):2287–2295. 1
- [16] T. Ju, F. Losasso, S. Schaefer, and J. Warren. Dual contouring of hermite data. In Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '02, pp. 339–346. ACM. event-place: San Antonio, Texas. doi: 10.1145/566570.566586 3
- [17] V. Khanduja, J. E. Lawrence, and E. Audenaert. Testing the construct validity of a virtual reality hip arthroscopy simulator. 33(3):566–571. 1
- [18] G. Kindlmann and J. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In IEEE Symposium on Volume Visualization (Cat. No.989EX300), pp. 79–86. doi: 10.1109/SVV.1998.729588 3
- [19] J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional transfer functions for interactive volume rendering. 8(3):270–285. doi: 10.1109/TVCG.2002.1021579 3
- [20] S. Kockara. Balls hierarchy: A proximity query for metric spaces and covering theorem. In 2009 International Conference on Network-Based Information Systems, pp. 512–518. ISSN: 2157-0418, 2157-0426. doi: 10.1109/NBiS.2009.100 3
- [21] S. Kockara, V. Yip, and M. Mete. Balls hierarchy: Image segmentation by graph spanner. In 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, pp. 514–517. IEEE. 3
- [22] R. J. Koehler, S. Amsdell, E. A. Arendt, L. J. Bisson, J. P. Braman, A. Butler, A. J. Cosgarea, C. D. Harner, W. E. Garrett, T. Olson, W. J. Warne, and G. T. Nicandri. The arthroscopic surgical skill evaluation tool (ASSET). 41(6):1229–1237. doi: 10.1177/0363546513483535 1
- [23] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In ACM siggraph computer graphics, vol. 21, pp. 163–169. ACM. 2, 3
- [24] D. Morris, C. Sewell, N. Blevins, F. Barbagli, and K. Salisbury. A collaborative virtual environment for the simulation of temporal bone surgery. In International conference on medical image computing and computer-assisted intervention, pp. 319–327. Springer. 2
- [25] A. Petersik, B. Pflesser, U. Tiede, K.-H. Höhne, and R. Leuwer. Realistic haptic interaction in volume sculpting for surgery simulation. In International Symposium on Surgery Simulation and Soft Tissue Modeling, pp. 194–202. Springer. 2
- [26] Y. Puljajala, M. Ma, M. Pears, D. Peebles, and A. Ayoub. Effectiveness of immersive virtual reality in surgical training—a randomized control trial. 76(5):1065–1072. 1, 2
- [27] S. Shedage, J. Farmer, D. Demirel, T. Halic, S. Kockara, V. Arikatla, K. Sexton, and S. Ahmadi. Development of virtual skill trainers and their validation study analysis using machine learning. In 2021 the 5th International Conference on Information System and Data Mining, pp. 8–13. ACM. doi: 10.1145/3471287.3471296 1
- [28] B. Stew and E. Ooi. The role of simulation in endoscopic sinus surgery

- training. p. 93. 1
- [29] R. Treuting. Minimally invasive orthopedic surgery: arthroscopy. 2(3):158–163. 1
- [30] M.-D. Tsai, M.-S. Hsieh, and C.-H. Tsai. Bone drilling haptic interaction for orthopedic surgical simulator. 37(12):1709–1718. Publisher: Elsevier. 2
- [31] K. R. Vaghela, A. Trockels, J. Lee, and K. Akhtar. Is the virtual reality fundamentals of arthroscopic surgery training program a valid platform for resident arthroscopy training? 480(4):807–815. Publisher: LWW. 1
- [32] N. Vaughan, V. N. Dubey, T. W. Wainwright, and R. G. Middleton. A review of virtual reality based training simulators for orthopaedic surgery. 38(2):59–71. 1
- [33] D. Wang, S. Zhao, T. Li, Y. Zhang, and X. Wang. Preliminary evaluation of a virtual reality dental simulation system on drilling operation. 26:S747–S756. 2
- [34] J. Wu, G. Yu, D. Wang, Y. Zhang, and C. C. Wang. Voxel-based interactive haptic simulation of dental drilling. In *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 39–48. American Society of Mechanical Engineers. 2, 5
- [35] A. Ziv. Patient safety and simulation-based medical education. 22(5):489–495. 1