Visuohaptic Simulation of Bone Surgery for Training and Evaluation

Dan Morris, Christopher Sewell, Federico Barbagli, and Kenneth Salisbury
Stanford University

Nikolas H. Blevins and Sabine Girod
Stanford University Medical Center

Visual and haptic simulation of bone surgery can support and extend current surgical training techniques. The authors present a system for simulating surgeries involving bone manipulation, such as temporal bone surgery and mandibular surgery, and discuss the automatic computation of surgical performance metrics. Experimental results confirm the system's construct validity.

Surgical training has traditionally revolved around an apprenticeship model: Residents observe experienced surgeons in the operating room and eventually are deemed ready to perform their first procedure.[1] Residents in craniofacial surgery, for example, learn anatomy primarily from textbooks and models, and surgical technique through apprenticeship and procedure observation.

Resident training in otologic surgery typically includes dissection of preserved human temporal bones. This lets residents become acquainted with the mechanical aspects of drilling, but doesn't incorporate physiological information, continuous feedback for hazard avoidance, or soft tissue work. Temporal bone labs are also costly to maintain, and cadaver specimens can be difficult to obtain in sufficient quantity. This approach also limits the precision with which an instructor can monitor a trainee's drilling performance, because the instructor can't feel the fine details of the trainee's interaction with the bone surface, and can't easily share the drill and bone surface for demonstration. Furthermore, instructors have little or no mechanism for controlling anatomic variations or the presence of specific pathology that can lead to challenging training scenarios.

In recent years, simulation-based training has emerged as a potential adjunct to traditional methods.[2] Simulation offers a safe, cost-effective, customizable, and easily accessible tool for gaining surgical experience. This article presents methods for simulating surgeries involving bone manipulation, with a specific focus on two categories of procedures: temporal bone surgery and mandibular surgery.

Several common otologic surgical procedures—including mastoidectomy, acoustic neuroma resection, and cochlear implantation—involve drilling within the temporal bone to access critical anatomy within the middle ear, inner ear, and skull base. As computer simulation has become a more frequently used technique in surgical training and planning, this class of procedures has emerged as a strong candidate for simulation-based learning. The time spent on a procedure in this area is typically dominated by bone removal, which is performed with a series of burrs (rotary drill heads) of varying sizes and surface properties. Larger burrs are generally used for gross bone removal in the early part of a procedure, while smaller burrs are used for finer work in the vicinity of target anatomy. Surgeons use a variety of strokes and contact techniques to precisely control bone removal while minimizing the risk of vibration and uncontrolled drill motion that could jeopardize critical structures.

Mandibular procedures are also likely to benefit from surgical simulation for several reasons. The complex, patient-specific planning process and the significant case-to-case anatomic variation suggest that an end-to-end simulator will assist physicians in preparing for specific cases. Furthermore, distraction procedures have been introduced to the craniofacial surgical community only within the last 10 to 15 years, and an effective simulator will significantly aid in the training and retraining of this new class of procedures, and with the exploration of alternative techniques for effective surgeries.

Simulation and rendering

Our simulation's goal is high-fidelity presentation of the visual and haptic cues present in a surgical environment. This section discusses our overall rendering scheme, focusing on how we present the specific cues that are relevant to surgical training.

Data sources and preprocessing

We load models from full-head or temporal bone computerized tomography (CT) data sets, threshold them to isolate bone regions, and resample them to produce isotropic voxels of 0.5 millimeters per side. Using a standard resampled resolution lets us calibrate our rendering approaches independently of the image sources used for a particular simulation case.

 Published by the IEEE Computer Society

Hybrid data structure generation

We maintain a hybrid data structure that uses volumetric data for haptic rendering and traditional triangle arrays for graphic rendering. This lets us leverage previous work on haptic rendering of volumetric data (see the "Previous Work in Temporal Bone Surgery Simulation" sidebar, next page) while maintaining the benefits of surface rendering in terms of hardware acceleration and visual effects. To simplify and accelerate the process of updating our polygonal data when the bone is modified, we build a new surface mesh in which vertices correspond directly to bone voxels, rather than use the original isosurface mesh.

We load the voxel array representing the bone model into our simulation environment, and generate a polygonal surface mesh to enclose the voxel grid by exhaustively triangulating the voxels on the bone region's surface. That is,

```
for each voxel v1
if v1 is on the bone surface
  for each of v1's neighbors v2
  if v2 is on the bone surface
    for each of v2's neighbors v3
    if v3 is on the bone surface
      generate vertices
        representing v1, v2, v3
      generate a triangle t(v1,
        v2, v3)
      orient t away from the bone
        surface
```

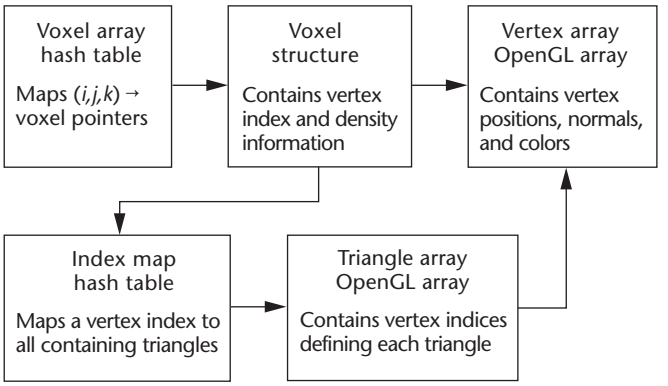
Here, we define being "on the bone surface" as having nonzero bone density and having at least one neighbor with no bone density. This generates a significant number of triangles (about 200,000 for a typical full-head CT data set). To avoid generating duplicate triangles, we assign each voxel an index before tessellation, and reject triangles if they don't appear in sorted order. A second pass over the mesh uses Bouvier's[3] observations to eliminate subsurface triangles that won't be visible from outside the mesh.

A compact, in-memory hash table, indexed by 3D grid coordinates, stores the voxels. This allows rapid point and volume collision-detection without excessive memory requirements.

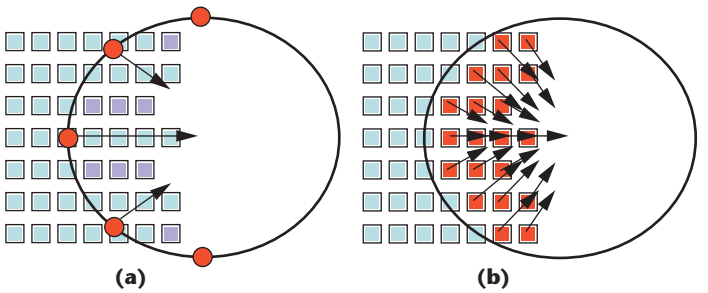
Secondary data structures map each voxel to its corresponding vertex index, and each vertex index to the set of triangles containing it. This allows rapid access to graphic-rendering elements (vertices and triangles) given a modified bone voxel, which is critical for shading vertices based on voxel density and for retriangulation when voxels are removed. Figure 1 summarizes the relevant data structures.

Haptic rendering

Users control virtual instruments with a SensAble Phantom[4] haptic feedback device, which provides 3-degree-of-freedom (DoF) force feedback and 6-DoF positional input. They can select from a variety of drills, including diamond and cutting burrs ranging from 1 to 6 millimeters in diameter.



1 Summary of the structures binding our volumetric (haptic) and surface (graphic) rendering data. When voxels are removed or modified, the corresponding vertices and triangles can be accessed from the (*i*, *j*, *k*) voxel index in approximately constant time.



2 Summary of contrasting approaches to haptic rendering. Red points are surface samples on a spherical drill's surface. (a) In the ray-tracing approach, each sample contributes a vector to the overall force that points toward the tool center and is proportional to the sample's penetration. The ray-tracing algorithm would miss voxels labeled in purple, creating uneven bone removal. (b) In our volume-sampling approach, the drill's full volume is sampled, and each point that's found to be immersed in the bone volume contributes a unit-length vector to the overall force that points toward the tool center.

Gross feedback: volume sampling. We initially adopted a haptic feedback approach similar to Petersik et al.'s,[5] in which the drill is represented as a cloud of sample points, distributed approximately uniformly around a spherical burr's surface. At each time step, the system tests each sample point for contact with bone tissue. By tracing a ray from each immersed sample point toward the tool's center, the system can generate a contact force that moves that sample point out of the bone volume (see Figure 2a).

Despite working well overall, this approach had several undesirable artifacts. Because of sampling effects (Figure 2a), the approach produced uneven voxel removal at high resolutions, creating unrealistic bone removal patterns that depended on surface sampling. Furthermore, floating-point computations are required to find the intersection points at which rays enter and leave voxels. Because sampling density is limited by the number of samples that can be processed in a haptic time step (approximately 1 millisecond), extensive floating-point computation limits the potential sampling density. This sparse

Previous Work in Temporal Bone Surgery Simulation

Previous work in interactive simulation of temporal bone surgery[1,2] has focused primarily on haptic rendering of volumetric data. Agus et al.[1] developed an analytical model of bone erosion as a function of applied drilling force and rotational velocity, which they verified with experimental data.[3] Pflesser et al.[2] and Petersik et al.[4] model a drilling instrument as a point cloud, and use a modified version of the Voxmap-Pointshell algorithm[5] to sample the drill's surface and generate appropriate forces at each sampled point. This work developed into a commercial simulator (see http://www.voxel-man.de/simulator/temposurg). Each of these projects incorporates haptic feedback into volumetric simulation environments that use computerized tomography and magnetic resonance data.

Agus et al.[1] describe several enhancements to their simulation environment that incorporate additional skills, including the use of irrigation and suction; and additional sources of intraoperative feedback, including real-time rendering of bone dust.

Additional work has focused on noninteractive simulation of craniofacial surgery for planning and outcome prediction.[6-8] Morris et al. discuss preliminary work on the interactive simulation of craniofacial surgery,[9] and Gibson et al. present a simulation architecture for arthroscopic procedures.[10]

References

1. M. Agus et al., "A Multiprocessor Decoupled System for the Simulation of Temporal Bone Surgery," *Computing and Visualization in Science*, vol. 5, no. 1, 2002, pp. 35-43.
2. B. Pflesser et al., "Volume Cutting for Virtual Petrous Bone Surgery," *Computer Aided Surgery*, vol. 7, no. 2, 2002, pp. 74-83.
3. M. Agus et al., "Physics-Based Burr Haptic Simulation: Tuning and Evaluation," *Proc. 12th IEEE Haptics Symp.*, IEEE CS Press, 2004, pp. 128-135.
4. A. Petersik et al., "Haptic Volume Interaction with Anatomic Models at Sub-Voxel Resolution," *Proc. IEEE Virtual Reality*, IEEE CS Press, 2002, pp. 66-72.
5. M. Renz et al., "Stable Haptic Interaction with Virtual Environments Using an Adapted Voxmap-Pointshell Algorithm," *Proc. Eurohaptics*, 2001, pp. 149-154; http://www.eurohaptics.vision.ee.ethz.ch/2001/renz.pdf.
6. E. Keeve et al., "Deformable Modeling of Facial Tissue for Craniofacial Surgery Simulation," *Computer Aided Surgery*, vol. 3, 1998, pp. 228-238.
7. R.M. Koch et al., "A Framework for Facial Surgery Simulation," *Proc. 18th Spring Conf. Computer Graphics*, ACM Press, 2002, pp. 33-42.
8. J.G. Schmidt et al., "A Finite Element Based Tool Chain for the Planning and Simulation of Maxillo-Facial Surgery," *Proc. 4th European Conf. Computational Fluid Dynamics* (ECCOMAS), Univ. of Jyväskylä, 2004, pp. 1-17.
9. D. Morris et al., "An Interactive Simulation Environment for Craniofacial Surgical Procedures," *Proc. Medicine Meets Virtual Reality XIII* (MMVR), IOS Press, 2005, pp. 334-341.
10. S.F. Gibson et al., "Simulating Arthroscopic Knee Surgery Using Volumetric Object Representations, Real-Time Volume Rendering and Haptic Feedback," *Proc. 1st Joint Conf. Computer Vision, Virtual Reality, and Robotics in Medicine and Medial Robotics and Computer-Assisted Surgery*, Springer, 1997, pp. 369-378.

sampling limits the simulation's effective stiffness (which depends on rapid and accurate computation of penetration volume), disrupting the illusion of contact with a highly rigid object. It also limits the implementation of certain higher-level effects, such as bone modification, which depend on the precise subparts of the drill that contact the bone.

We take a more exhaustive approach to sampling the tool for haptic feedback and bone density reduction. We discretize the tool itself into a voxel grid (generally at a finer resolution than the bone grid), and a preprocessing step computes an occupancy map for the tool's voxel array. At each interactive time step, we check each volume sample in the tool for intersection with the bone volume (a constant-time, integer-based operation, using the hash table described earlier). A sample point inside a bone voxel generates a unit-length contribution to the overall haptic force vector that tends to push this sample point toward the tool center, which—with adequate stiffness—is always outside the bone volume (Figure 2b). Thus, we compute overall penetration depth based on the number of immersed sample points, rather than on the results of a per-sample ray trace.

The overall force generated by our approach is thus oriented along a vector that is the sum of the contributions from individual volume sample points. This force's magnitude increases with the number of sample points immersed in the bone volume.

Nonlinear magnitude computation. Because the drill is densely sampled, numerous sample points often become immersed immediately after the drill surface penetrates the bone volume, leading to instability during low-force contact. Reducing the overall stiffness leads to softer haptic feedback that doesn't accurately represent bone stiffness. We thus use a multigain approach, in which the haptic feedback's magnitude is a nonlinear function of the number of immersed sample points.

More specifically, we define two gains: one for shallow penetrations (when fewer than a threshold number of sample points are immersed); the other for deeper penetrations. We set this threshold such that the discontinuity in the force function occurs shortly after contact is initiated, so the user perceives no discontinuity. This approach allows large stiffnesses during haptic interaction, while avoiding instability during the high-risk period immediately following initial penetration.

Our volume-sampling approach requires sampling significantly more points than the ray-tracing approach because we sample the burr's complete volume, not just its surface. However, the operation performed when a tool sample lies within the bone volume is a constant-time computation, rather than a complex ray-tracing operation. Overall, we can achieve a significantly higher stiffness than the ray-tracing approach allows. We build on the ray-tracing approach for less-time-critical tasks, including bone thickness estimation and haptic feedback for nonphysically based tools.

Modeling drill surface nonuniformity. Our system associates a drilling power with each sample point based on its location in the drill head. Each tool voxel that intersects a bone voxel removes an amount of bone density that depends on the sample point's drilling power. This approach lets us simulate key aspects of drill and bone contact, particularly the fact that the burr's equatorial surface carries a larger linear velocity than

its polar surface and thus removes more bone per unit of applied force. Simulating this effect is critical for encouraging trainees to use a proper drilling technique.

More precisely, the amount of bone removed per time unit by a given sample point is computed as R_{br} in the following expression:

$$\theta = \text{abs}(\cos^{-1}(\mathbf{d} \cdot (\mathbf{s} - \mathbf{t_c})))$$
$$R_{br} = f \cdot \max(0, R_{\max} - falloff \cdot \text{abs}((\pi/2) - \theta))$$

where $\mathbf{s}$ is the sample point's location, $\mathbf{t_c}$ is the tool center's location, $\mathbf{d}$ is the tool handle's axis, and θ is the angle between the drill handle and $(\mathbf{s} - \mathbf{t_c})$. The expression $\text{abs}(\pi/2 - \theta)$ is the current sample point's latitude, and *falloff* is a constant parameterizing the drill surface's nonuniformity. If *falloff* is zero, the drill's pole and equator remove bone with equal efficiency. $R_{\max}$ is the maximum rate of bone removal per unit force, and f is the magnitude of force the user is currently applying. Figure 3 summarizes the computation of latitude. We precompute *falloff* parameters for drill samples to avoid performing expensive arc-cosine operations hundreds of times per haptic time step.

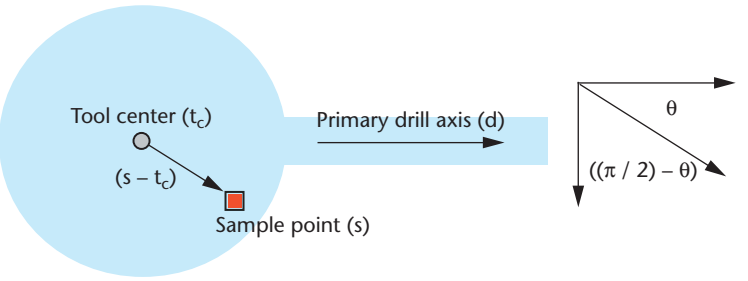
This approach encourages proper drilling technique and lets us model critical differences among burr types. For example, our model captures the fact that cutting burrs typically depend on drilling angle more than diamond burrs do, but have higher overall bone removal rates. A cutting burr would thus be associated with both a higher $R_{\max}$ and a higher *falloff* in the previous expression.

Modeling tangential forces. Another property of surgical drills that should be accurately represented in a simulation environment is their tendency to drag the user along the bone's surface because of the contact forces between the drilling burr's teeth and the bone. Stroking the drill on the bone surface in a direction that lets these forces oppose a surgeon's hand motion permits the surgeon to control the drill's velocity. Stroking the drill such that these forces complement the surgeon's hand motion causes the drill to catch its teeth on the bone and rapidly run in the movement's direction, which can be extremely dangerous. Simulating this effect is thus critical to training correct drilling technique.

Modeling the contact forces between the individual teeth in the drill's geometry and the bone surface would be computationally expensive, so we again use our dense sampling approach to approximate tangential drill forces during penalty force computation.

Each sample found to be immersed in the bone (for example, the red samples in Figure 2b) computes its own tangential force vector, according to $f_{\tan} = (\mathbf{p} - \mathbf{sc}) \times \mathbf{d}$, where $f_{\tan}$ is the tangential force created by this sample, $\mathbf{p}$ is the sample's position, $\mathbf{sc}$ is the center of the drill slice in which this sample lies (the sample position projected onto the drill axis), and $\mathbf{d}$ is the drill's primary axis (and thus the axis of rotation), as Figure 3 shows.

The vector $(\mathbf{p} - \mathbf{sc})$ from the tool axis to this sample point is an approximation of the local surface normal (the true surface normal is generally unknown, because most samples aren't on the model's surface and thus don't have defined normals). The drill axis vector is normalized to unit length, and the magnitude of the vector $(\mathbf{p} - \mathbf{sc})$ indicates its distance from the tool axis and thus its linear velocity (because the drill spins at constant rotational velocity, samples farther from the rotation axis carry larger linear velocity than those near it). The cross-product $(\mathbf{p} - \mathbf{sc}) \times \mathbf{d}$ is thus scaled according to sample velocity, and is perpendicular to both the drill's axis and the approximate surface normal.



3 Computing the latitude of a volume sample point for bone removal rate computation.

Summing these vectors over all samples that lie on the bone creates a net force that simulates the interaction between the drill's teeth and the bone surface. Scaling this vector by –1 is equivalent to reversing the drill's handedness.

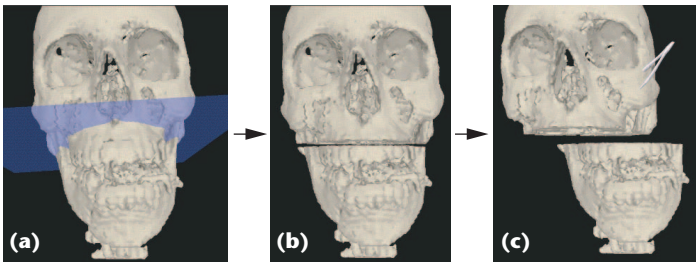
Modeling drill vibration using recorded data. Another key aspect of the haptic sensation associated with drilling is the instrument's vibration, which varies with applied force and burr type. To generate realistic drill vibration frequencies, we outfitted a physical drill with an accelerometer and collected vibration data at a variety of applied drilling forces.

We identified the key spectral peaks for each burr type and used them to synthesize vibrations during the simulation. Because we drive our haptic feedback device at approximately 1.5 kHz, we can't preserve the highest-frequency vibrations identified in these experimental recordings. However, we can preserve the lower-frequency harmonics and the variations in vibration associated with changes in burr type and applied drilling force.

Data manipulation

When bone voxels are removed from our environment, our hybrid data structure requires that we retessellate the area around the removed bone. Consequently, our haptic rendering thread queues bone voxels as they're removed, and the graphic rendering thread retessellates the region around each voxel pulled from this queue. That is, for each removed voxel, we determine which of its neighbors have been revealed and create triangles containing the centers of these new voxels as vertices. Specifically, for each removed voxel v, we perform the following steps:

```
for each voxel v' that is adjacent to v
  if v' is on the bone surface
    if a vertex has not already been
       created to represent v'
      create a vertex representing v'
```



4 Use of the cut-plane tool and independent manipulation of discontinuous bone regions. (a) With the cut-plane tool, the user geometrically specifies a set of voxels to remove. (b) The volume after voxel removal. (c) The flood-filling thread recognizes the discontinuity, and the user can now manipulate the bone segments independently.

```
        compute the surface gradient at v'
        queue v' for triangle creation

  for each queued voxel v'
    generate triangles adjacent to v'
```

Once again, a voxel is defined to be on the bone surface if it has a nonzero bone density and at least one neighboring voxel containing no bone density. When we've tested all local voxels for visibility (that is, when the first loop is complete in the previous pseudocode), we feed all new vertices to a triangle-generation routine. This routine finds new triangles that can be constructed from new vertices and their neighbors, orients those triangles to match the vertices' surface normals, and copies visible triangles to the visible triangle array. We queue triangles for triangle creation because triangle generation (performed in the second loop of the pseudocode) depends on knowing which local voxels are visible, something possible only after the first loop's completion.

Additional tools

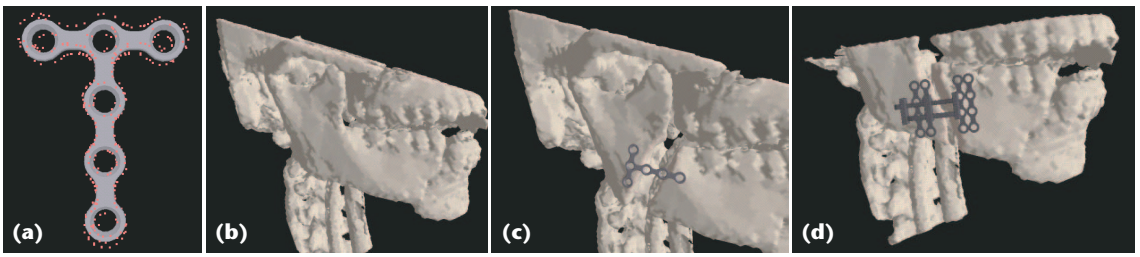
An additional bone-modification tool lets us introduce large bone cuts via a planar cut tool (see Figure 4). This tool isn't intended to replicate a physical tool. Rather, it addresses the need of advanced users to make rapid cuts for demonstration or for creating training scenarios. We implement bone removal with this tool by discretizing the planar area—controlled in 6 DoF—into voxel-sized sample areas, and tracing a ray a small distance from each sample along the normal to the plane. This is similar to Petersik et al.'s[5] haptic rendering approach, but we employ this approach without generating haptic feedback, and each ray is given infinite drilling power—that is, we remove all density from any voxels through which each ray passes. The user controls the distance traced along each ray. This lets the user remove a planar or box-shaped region of bone density, as Figure 4b demonstrates. This approach will often generate isolated bone fragments that the user will want to move or delete.

A final set of tools lets users manipulate rigid models that can be bound to bone objects. This is particularly relevant for the target craniofacial procedures, which center on rigidly affixing metal plates to the patient's anatomy. We thus provide models of several distractors (instruments used to slowly separate bone fragments postoperatively) and industry-standard bone plates (adding more models is straightforward). Including these plate models lets users plan and practice plate-insertion operations interactively. We perform collision detection for haptic feedback using a set of sample points, as we did with drilling tools. In this case, we generate the sample points by sampling 100 vertices of each model and extruding them slightly along their normals (because these models tend to be thin relative to our voxel dimensions), as Figure 5a shows. To handle bone contact for this tool, which generally involves objects with much larger volumes than the drill tools, we use ray tracing.[5] This approach allows reasonable haptic feedback with fewer samples than the volumetric approach we use for our drilling tools. Because there's no well-defined tool center toward which we can trace rays for penetration calculation, we trace rays along the model's surface normal at each sample point. At any time, the user can rigidly affix a plate tool to a bone object with which it's in contact using a button on the haptic device (see Figures 5b, 5c, and 5d).

Discontinuity detection

A critical step in simulating craniofacial procedures is detecting cuts in the bone volume that separate one region of bone from another, thus letting us apply independent rigid transformations to the isolated bone segments.

In our environment, a background thread performs a repeated flood-filling operation on each bone structure. A random voxel is selected as a seed point for each bone object, and flood-filling proceeds through all voxel neighbors that currently contain bone density. Each voxel maintains a flag indicating whether the flood-fill-



5 Modeling and attaching rigid bone plates. (a) A bone plate's surface after sampling and extrusion. (b) A bone surface before modification. (c) The same bone surface after drilling, distraction, and plate attachment. (d) The same bone surface after drilling, distraction, and distractor insertion.

ing operation has reached it. At the end of a filling pass, the thread collects all unmarked voxels (which must have been separated from the seed point) and moves them into a new bone object, along with their corresponding data in the vertex and triangle arrays.

Figures 4a and 4c show a bone object that has been cut and the subsequent independent movement of the two resulting structures. For demonstration, we use the cut-plane tool to create the fracture; during simulated procedures, fractures are generally created by the drilling and sawing tools.

Graphic rendering

To exploit the fact that the user doesn't frequently change the simulation's viewing perspective, we maintain two triangle arrays—one containing the complete tessellation of the current bone volume (the complete array), and one containing only those visible from positions close to the current camera position (the visible array). We initialize the latter array at start-up and reinitialize it any time the camera comes to rest after a period of movement. Visible triangles have at least one vertex whose normal points toward (less than 90 degrees away from) the camera. Because this visibility testing pass is time-consuming, the system performs it in the background. We use the complete array for rendering the scene during periods of camera movement (when the visible array is considered dirty) and during the visible array's reinitialization.

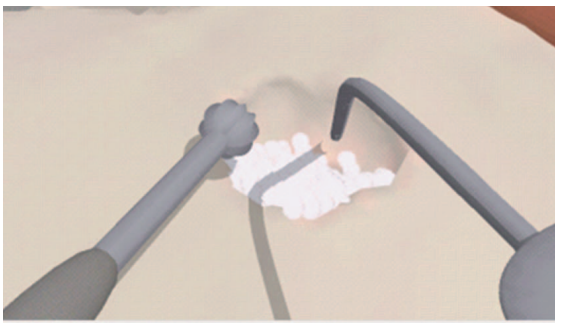
As an additional optimization, we use the nvtristrip library (see http://developer.nvidia.com) to reorder our triangle and vertex arrays for optimal rendering performance. We could have further reduced rendering time by generating triangle strips from our triangle lists, but this would add significant computational complexity to the time-critical process of updating the surface mesh to reflect changes to the underlying voxel grid.

Bone dust simulation

We also build on Agus et al.'s work[6] to provide a simulation of bone dust accumulation, which is particularly critical in otologic procedures. Bone dust tends to accumulate in the drilling area, and must be suctioned off to enhance visibility of the bone surface.

Agus et al.[6] simulate the behavior of individual particles of bone dust, sampling a subset of the particles in each rendering pass to minimize the computational load demanded by the simulation. Because individual particles of bone dust aren't generally visible, we don't need to simulate particulate motion. We therefore take an Eulerian approach similar to Stam,[7] in which we discretize the working region into a 3D hashed grid. Rather than tracking individual particles, we track the density of particles contained in each grid cell. This lets us simulate the piling of dust particles, particle flow due to gravity, and particle movement due to tool contact for all accumulated bone dust, without simulating individual particles. Gravity and tool forces transfer density between neighboring grid cells, rather than modifying the velocity of individual particles.

We render each grid cell containing bone dust as a partially transparent OpenGL quad, whose dimensions scale with the density of dust in that cell. This gives a convincing representation of accumulated particle volume and density, and doesn't require that we render each particle (that is, each quantum of density) individually.



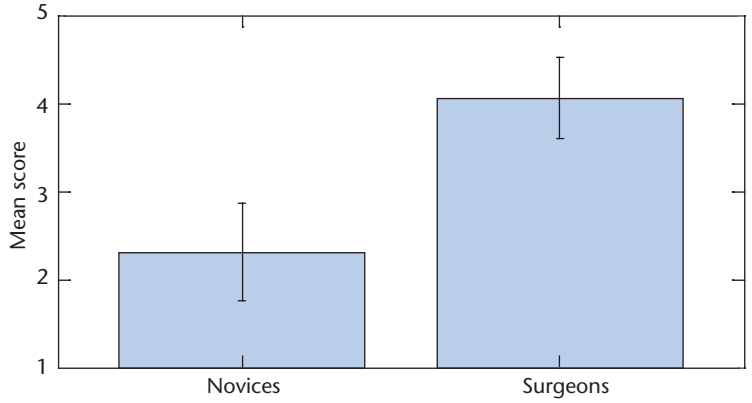
6 Bone dust simulation. The user has removed a volume of bone, which has now accumulated as bone dust. The physical simulation has allowed the bone dust to fall to the bottom of the drilled area. The user is preparing to remove the bone dust with the suction device.

This grid-based approach significantly reduces computation and rendering time relative to a particle-based (Lagrangian) approach. Coupled with the hash table we use to minimize memory consumption for the grid, this approach lets us render large quantities of accumulated bone dust without impacting the application's interactive performance. Figure 6 shows a volume of accumulated bone dust and the suction device the trainee uses to remove it. The suction device is controlled with an additional Phantom haptic interface.

Data-driven sound synthesis

Sound is a key source of intraoperative feedback, providing information about drill contact and the nature of the underlying bone. We simulate the virtual burr's sound as a series of noisy harmonics, whose frequency modulates with the applied drilling force. Building on Bryan et al.'s harmonic-based synthesis approach,[8] we've recorded audio data from cutting and diamond drill burrs applied to cadaver temporal bone under a series of drilling forces to determine the appropriate frequencies for synthesized sound, as well as the dependence of this data on drill type and applied drilling force.

Sound can also be a key indicator of bone thickness intraoperatively. Sound quality and frequency change significantly as the drill contacts a thin layer of bone to warn that the surgeon is approaching sensitive tissue. In our simulator, the synthesized sound's pitch increases when the drilled area becomes thin. To estimate the thickness of bone regions, we used a ray-tracing algorithm similar to Petersik et al.'s algorithm for haptic rendering.[5] At each voxel determined to be on the bone's surface, we use the surface gradient to approximate the surface normal, and cast a ray into the bone along this normal. We trace the ray until it emerges from the bone volume, and estimate the thickness as the distance from the ray's entry point to its exit point. For sound synthesis, we average this thickness over all surface voxels with which the drill is in contact. Below an empirically selected thickness threshold, sound frequency increases linearly with decreasing bone thickness. We select this relationship's slope so that the key harmonics span the same range of frequencies in simulation that they do in our measured data.



7 **Mean scores for simulated mastoidectomies performed by novice participants (left) and participants with surgical experience (right). Error bars indicate 95 percent confidence intervals.**

Results: construct validity

The surgical simulation community defines several levels of validity—that is, a simulator's ability to mimic the real-world properties of the environment it aims to represent. Our study assesses the construct validity of our simulation environment—that is, the ability to explain subject behavior in simulation with appropriate parameters describing subject experience level. In other words, expert surgeons should perform objectively better on a simulated surgical task than novices.

We asked 15 right-handed participants to perform a mastoidectomy (removal of a portion of the temporal bone and exposure of relevant anatomy) in our simulator. Participants included four experienced surgeons, four residents in head and neck surgery with surgical experience, and seven novices with no surgical experience.

We presented participants with a tutorial of the simulator and gave them 15 minutes to practice using the haptic devices and the simulator's user interface. We then showed them an instructional video describing the target procedure, and gave them access—before and during the procedure—to still images indicating the desired appearance of the bone model at various stages in the procedure. We asked participants to perform the same procedure twice.

We logged each participant's hand movements, haptic forces, and surgical interactions to disk, and later rendered them to video. An experienced head and neck surgery instructor scored videos on a scale of 1 to 5; the instructor didn't know which videos came from which subjects and viewed them in randomized order. This global scoring approach is similar to the approach used to evaluate resident progress in a cadaver training lab. Our hypothesis is that participants with surgical experience should receive consistently higher scores than those with no surgical experience.

Figure 7 summarizes the experimental results. Participants with surgical experience received a mean score of 4.06, and novices received a mean score of 2.31, a statistically significant difference according to a one-tailed t-test ($p < 0.0001$). This clear difference in performance when operating in our simulator demonstrates the system's construct validity.

Novel training techniques

Our simulators let us not only replicate interaction with bones—that is, replicate features available in a traditional cadaver-based training lab—but also incorporate training features that aren't possible in a traditional training lab. Thus, simulation has the potential not only to replicate but also to extend existing training techniques.

Haptic tutoring

Surgical training typically focuses on visual observation of experienced surgeons and verbal descriptions of proper technique. It's impossible for a surgeon to physically demonstrate the correct feel of bone manipulation with physical tools. With that in mind, we've incorporated a haptic mentoring module into our environment, letting trainees experience forces that result from a remote user's interaction with the bone model.

Ideally, the trainee would experience both the instructor's tool movements and the force applied by the instructor, but it's difficult to control both the position and the force at a haptic end-effector without any control of the compliance of the user's hand. To address this issue, we bind the trainee's tool position to that of an instructor's tool (running on a remote machine) via a low-gain spring, and add the resulting forces to a playback of the forces generated at the instructor's tool, according to

$$F_{\text{trainee}} = K_p(P_{\text{trainee}} - P_{\text{instructor}}) + F_{\text{instructor}},$$

where $F_{\text{instructor}}$ and F_{trainee} are the forces applied to the instructor's and trainee's tools, and $P_{\text{instructor}}$ and P_{trainee} are the position of the instructor's and trainee's tools. K_p is small enough that it doesn't interfere significantly with the perception of the high-frequency components transferred from the instructor's tool to the trainee's tool, but large enough that the trainee's tool stays in the vicinity of the instructor's tool. In practice, the error in this low-gain position controller is still within reasonable visual bounds, and the trainee perceives that he or she is experiencing the same force and position trajectory as the instructor.

We use the same approach and force constants for haptic playback, letting users play back force data collected from a previous user's run through our system. This has potential value both for letting trainees experience the precise forces applied during a canonically correct procedure, and for letting instructors experience and evaluate the precise forces generated during a trainee's trial run.

Neurophysiology console simulation

Another goal of our simulation environment is to train surgeons to avoid critical or sensitive structures when using potentially dangerous tools. The inferior alveolar nerve, for example, is at particular risk during most of the craniofacial procedures this environment is targeting. We thus incorporate a virtual nerve monitor that represents the activity of nerve bundles in the procedure's vicinity (Figure 8). Nerves are currently placed explicitly for training scenarios; future work will include automatic segmentation of large nerves from image data.

This approach might also contribute to the simula-

tion-based training of a complete surgical team, which often involves several technicians focused on neurophysiology monitoring. Simulated neural data is streamed out via Ethernet for remote monitoring, and can be visualized on a console similar to what would be available intraoperatively to a technician.

Automated evaluation and feedback

Another exciting possibility for virtual surgery is the use of simulation environments to automatically evaluate a trainee's progress and provide targeted feedback to help improve a user's surgical technique.

A straightforward approach to evaluating a trainee's performance on the simulator is determining whether a given objective has been achieved while avoiding injury to vulnerable structures (such as nerves, ossicles, or veins). However, many of the finer points of technique are taught not because failure to adhere to them will necessarily result in injury, but because it increases the likelihood of injury. Therefore, it's useful to be able to quantify the risk inherent in the trainee's performance.

We describe several metrics for evaluating a user's bone-drilling technique. We present approaches to both visualize and validate these metrics (confirming that they are medically meaningful).

Visibility testing

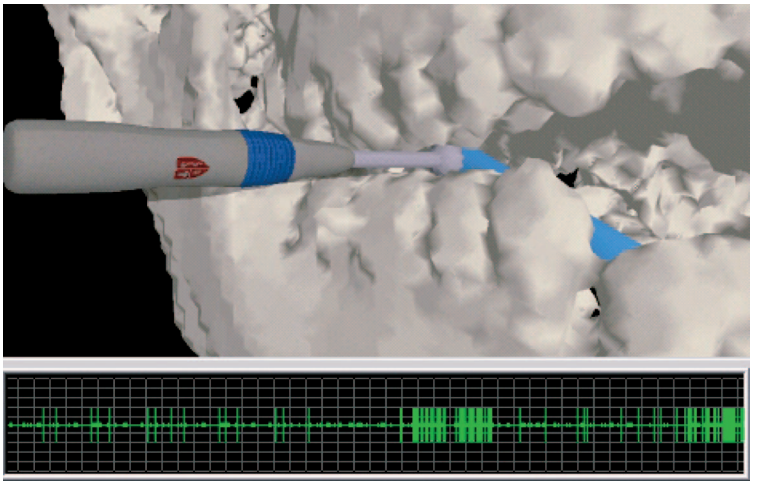
One of the most important ways to minimize risk in temporal bone surgery is to only remove bone that is within the line of sight. A saucerizing drilling technique (removing bone to create a saucer-shaped cavity on the bone surface) lets the surgeon avoid vulnerable structures just below the bone surface, using subtle visual cues that indicate their locations. Removing bone by undercutting (drilling beneath a shelf of bone that obscures visibility) increases risk of structure damage.

In our environment, as a user removes each voxel of bone, the simulator determines whether this voxel was visible to the user at the time of removal. Using the same ray-tracing techniques used for haptic rendering, the system traces a line from the removed voxel to the virtual eye point. If this ray intersects any voxels (other than those currently in contact with the drill), the removed voxel is determined to be invisible.

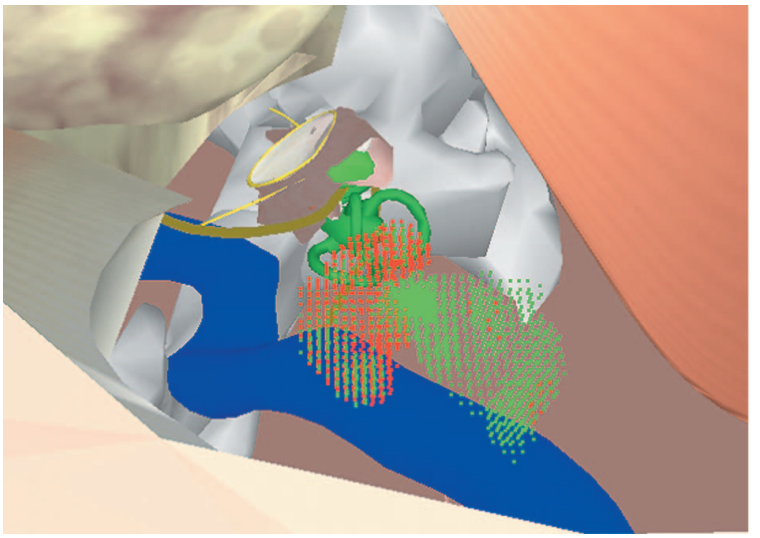
During or after a virtual procedure, a user can visualize the visibility or invisibility of every voxel he or she removed to explore the overall safety of the technique and find specific problem areas. Voxels that were visible when removed appear in one color, while those that were obscured are rendered in another color (see Figure 9). The scene might also be rotated and rendered with only selected structures visible, allowing unobstructed visualization of the removed voxels' locations and their proximities to crucial structures.

Although it makes intuitive sense that voxel visibility should be an appropriate metric for evaluating a user's performance, it's important to validate this metric—and all automatic metrics—against a clinically standard assessment of user performance. In this case, we use the data collected from the user study discussed in the "Results" section, which includes complete simulated procedures by experts and novices, along with scores assigned to each simulated procedure by an experienced surgical instructor. A metric that correlates well to an instructor's manually assigned scores will likely be an effective metric for automatic user evaluation.

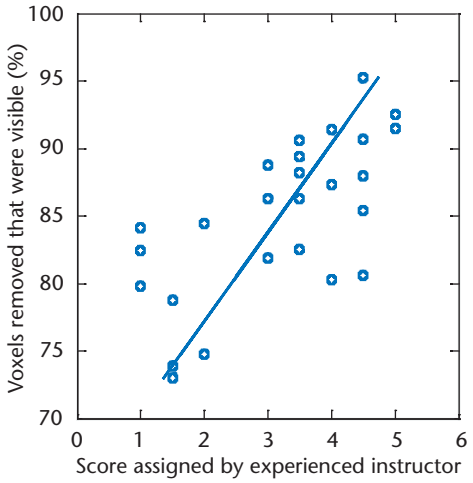
Figure 10 on the next page shows the results of correlating computed voxel visibilities to an instructor's score (on a scale of 1 to 5) for each simulated procedure performed by our study participants. Linear regression shows a correlation coefficient of 0.68, which is particularly high considering that we based the manual evaluation on a wide array of factors, only one of which was voxel visibility. This approach is suitable for assessing the effectiveness of individual metrics, which we can combine to form an overall score for a simulated procedure.



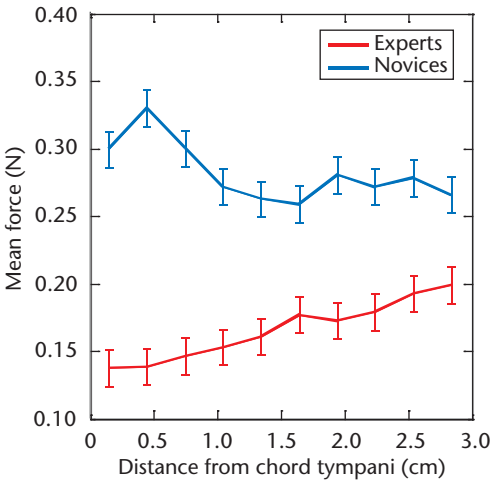
8 **Virtual neurophysiology monitoring. The user drills near a simulated nerve (in blue) and views a real-time simulated neural monitor, which also provides auditory feedback.**



9 **Visualization of removed voxels. This interactive visualization—in which the bone itself isn't rendered—displays the regions in which the trainee exercised proper technique (visible voxels in green) and regions in which he didn't (obscured voxels in red). Undercutting in close proximity to the sigmoid sinus (in blue) was dangerous because the trainee couldn't see the visual cues indicating the vein's location below the bone surface.**



10 Relationship between expert-assigned scores (*x*-axis) and computed voxel visibility (*y*-axis), along with a linear fit ($R = 0.68$, $p < 0.001$). Each dot represents one pass through the simulated procedure by one subject. The strong correlation supports the value of computed visibility as an automatic performance metric.

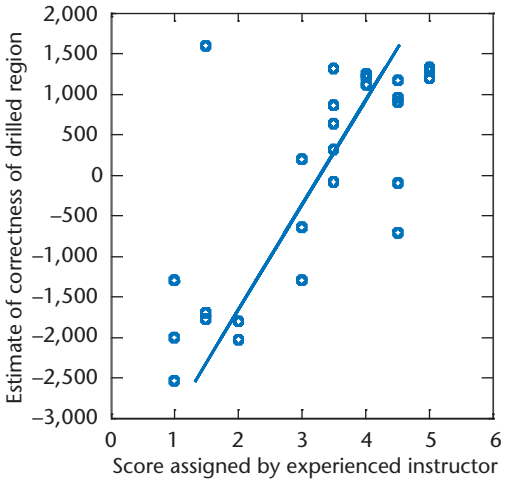


11 Forces applied by experts and novices in the vicinity of the chorda tympani (a sensitive branch of the facial nerve). Error bars indicate 95 percent confidence intervals.

Learning safe forces

Another component of safe drilling is applying appropriate forces and operating the drill at appropriate speeds. The acceptable range of forces and speeds is closely related to the drill's distance from vulnerable structures. However, this function is difficult for a human, even an expert surgeon, to quantify precisely. Therefore, we learn maximal safe forces and speeds via statistical analysis of forces, velocities, and distances recorded during a run of the simulation by experienced surgeons. We can then compare trainees' performance to the experts' values, and visualize areas in which users applied excessive speeds or forces.

For example, Figure 11 shows the force profiles of all expert and novice study participants as they approached a critical and sensitive structure (the chorda tympani, a branch of the facial nerve). At the instant that the user removed any voxel within 3 centimeters of this structure, the system recorded the user's applied force. We sorted these samples by distance from the nerve and binned them into 0.2-cm intervals. Figure 11 shows the mean value of each bin. The profiles for experts and novices are significantly different, as the plotted confidence intervals indicate. Experts clearly tend to use lower forces overall in the vicinity of this critical structure, reducing their forces as they approach, a trend not seen in the novice plots.



12 Relationship between expert-assigned scores (*x*-axis) and estimate of drilled region correctness (*y*-axis), along with a linear fit ($R = 0.76$, $p < 0.001$). Each dot represents one pass through the simulated procedure by one subject. The strong correlation supports the validity of our drilled-region-correctness estimate as an automatic performance metric.

Learning correct bone regions for removal

In addition to instantaneous metrics like force and visibility, an instructor evaluating a surgical trainee would also evaluate the overall shape of the drilled region after a complete procedure—that is, the set of voxels the trainee removed.

To capture this important criterion in a quantitative metric, we use a naive Bayes approach to categorize correct and incorrect drilling regions. We assume that voxels from the full voxel mesh are chosen for removal (drilling) according to separate distributions for experts and novices. For each voxel, we compute the probability that an expert would remove this voxel and the probability that a novice would remove it. Then, for each subject's run through a simulated procedure, we look at the set of removed voxels and determine the probability that an expert (or novice) performed the procedure by multiplying the probabilities of each removed voxel. We then compute the ratio of these cumulative probabilities (p_{expert} and p_{novice}) and take the log of that ratio to compute a scalar value that estimates the correctness of the drilled region ($\log(p_{expert}/p_{novice})$).

We'd like to show that this is a valid performance metric by correlating it with scores assigned by an experienced instructor. Figure 12 shows the result of this

analysis, along with a linear regression onto the scores assigned by an instructor ($R = 0.76$). Again, the high correlation suggests that this is a valuable component in a suite of individual metrics that can produce an accurate estimate of trainee performance.

Conclusion and future work

Subsequent work on the simulation environment will focus on incorporating a representation of soft tissue simulation into our environment. This will let us represent more complete procedures, including, for example, skin incision and tumor resection.

Subsequent work on our automated evaluation techniques will focus on the development of additional automated metrics and the visualization of automated metrics.

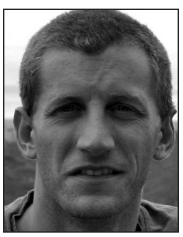
Supplemental material for this article, including movies and images of the simulation environment, is available at http://cs.stanford.edu/~dmorris/bonesim. ■

Acknowledgments

Support was provided by the US National Institute of Health under grant LM07295, BioX 2DMA178, the Association for Osteosynthesis, and National Defense Science and Engineering Graduate and Stanford Graduate Fellowships.

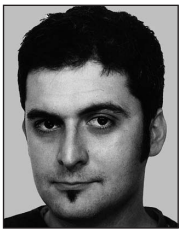
References

1. P.J. Gorman et al., "The Future of Medical Education Is No Longer Blood and Guts, It Is Bits and Bytes," *Am. J. Surgery*, vol. 180, no. 5, 2000, pp. 353-356.
2. R.S. Haluck et al., "Are Surgery Training Programs Ready for Virtual Reality?" *J. Am. College of Surgery*, vol. 193, no. 6, 2001, pp. 660-665.
3. D.J. Bouvier, "Double-Time Cubes: A Fast 3D Surface Construction Algorithm for Volume Visualization," *Proc. Int'l Conf. Imaging Science, Systems, and Technology*, 1997.
4. T.H. Massie and J.K. Salisbury, "The Phantom Haptic Interface: A Device for Probing Virtual Objects," *Proc. ASME Winter Ann. Meeting*, vol. 55, no. 1, 1994, pp. 295-300.
5. A. Petersik et al., "Haptic Volume Interaction with Anatomic Models at Sub-Voxel Resolution," *Proc. IEEE Virtual Reality*, IEEE CS Press, 2002, pp. 66-72.
6. M. Agus et al., "A Multiprocessor Decoupled System for the Simulation of Temporal Bone Surgery," *Computing and Visualization in Science*, vol. 5, no. 1, 2002, pp. 35-43.
7. J. Stam, "Real-Time Fluid Dynamics for Games," *Proc. Game Developer Conf.*, 2003; http://www.dgp.toronto.edu/people/stam/reality/Research/pdf/GDC03.pdf.
8. J. Bryan et al., "Virtual Temporal Bone Dissection: A Case Study," *Proc. IEEE Visualization*, Ertl et al., eds., IEEE CS Press, 2001, pp. 497-500.



Dan Morris *is a researcher at Microsoft Research. His research interests include interactive physical simulation, haptics, and neural prosthetics. Morris has a PhD in computer science from Stanford University. Contact him at dmorris@cs.stanford.edu.*

Christopher Sewell *is pursuing a PhD in computer science at Stanford. His research interests include surgical simulation, haptics, and machine learning. He has an MS in computer science from Stanford. Contact him at csewell@robotics.stanford.edu.*



Federico Barbagli *is a research fellow in the Robotics Laboratory at Stanford. His research interests include haptic rendering algorithms, haptic control, and haptic device design. Barbagli has a PhD in robotics and control from Scuola Superiore S. Anna. Contact him at barbagli@robotics.stanford.edu.*

Kenneth Salisbury *is a professor in the computer science and surgery departments at Stanford. His research interests include human–machine interaction, collaborative computer-mediated haptics, and surgical simulation. Salisbury has a PhD in mechanical engineering from Stanford. Contact him at jks@robotics.stanford.edu.*

Nikolas H. Blevins *is an assistant professor in the Department of Otolaryngology at Stanford. His research interests include surgical simulation, cochlear microendoscopy, microrobotics, and innovations in surgical education and preoperative planning. Blevins has an MD from Harvard and completed his residency in otolaryngology at the University of California at San Francisco. Contact him at nblevins@stanford.edu.*

Sabine Girod *is an assistant professor in the Division of Plastic Surgery at Stanford. Her clinical and research interests are functional oral and craniofacial rehabilitation of craniofacial trauma and craniofacial deformities, bone grafting, implantology, and oral pathology in children and adults. Girod has an MD from Hanover Medical School, Germany, where she also completed her residency in oral surgery. She has a PhD in molecular biology from the University of Cologne, Germany. Contact her at sgirod@stanford.edu.*