Old Dominion University

ODU Digital Commons

Electrical & Computer Engineering Faculty Publications

Electrical & Computer Engineering

2012

Procedural Wound Geometry and Blood Flow Generation for Medical Training Simulators

Rifat Aras
Old Dominion University

Yuzhong Shen
Old Dominion University, yshen@odu.edu

Jiang Li
Old Dominion University, jli@odu.edu

David R. Holmes III (Ed.)

Kenneth H. Wong (Ed.)

Follow this and additional works at: https://digitalcommons.odu.edu/ece_fac_pubs

 Part of the Biomedical Commons, Emergency Medicine Commons, Graphics and Human Computer Interfaces Commons, and the Medical Education Commons

Original Publication Citation

Aras, R., Shen, Y., & Li, J. (2012). Procedural wound geometry and blood flow generation for medical training simulators. In D.R. Holmes III & K.H. Wong (Eds.), *Medical Imaging 2012: Image-Guided Procedures, Robotic Interventions, and Modeling, Proceedings of SPIE Vol. 8316* (831622). SPIE of Bellingham, WA. https://doi.org/10.1117/12.911264

This Conference Paper is brought to you for free and open access by the Electrical & Computer Engineering at ODU Digital Commons. It has been accepted for inclusion in Electrical & Computer Engineering Faculty Publications by an authorized administrator of ODU Digital Commons. For more information, please contact digitalcommons@odu.edu.

Procedural wound geometry and blood flow generation for medical training simulators

Rifat Aras[a], Yuzhong Shen*[a], Jiang Li[b]
[a]Dept. of Modeling, Simulation, and Visualization Eng., Old Dominion University
[b]Dept. of Electrical and Computer Eng., Old Dominion University

ABSTRACT

Efficient application of wound treatment procedures is vital in both emergency room and battle zone scenes. In order to train first responders for such situations, physical casualty simulation kits, which are composed of tens of individual items, are commonly used. Similar to any other training scenarios, computer simulations can be effective means for wound treatment training purposes. For immersive and high fidelity virtual reality applications, realistic 3D models are key components. However, creation of such models is a labor intensive process. In this paper, we propose a procedural wound geometry generation technique that parameterizes key simulation inputs to establish the variability of the training scenarios without the need of labor intensive remodeling of the 3D geometry. The procedural techniques described in this work are entirely handled by the graphics processing unit (GPU) to enable interactive real-time operation of the simulation and to relieve the CPU for other computational tasks. The visible human dataset is processed and used as a volumetric texture for the internal visualization of the wound geometry. To further enhance the fidelity of the simulation, we also employ a surface flow model for blood visualization. This model is realized as a dynamic texture that is composed of a height field and a normal map and animated at each simulation step on the GPU. The procedural wound geometry and the blood flow model are applied to a thigh model and the efficiency of the technique is demonstrated in a virtual surgery scene.

Keywords: procedural techniques, wound geometry, surface blood flow, medical training simulators

1. INTRODUCTION

High fidelity 3D objects are the key components of immersive virtual reality applications. Unfortunately, the traditional modeling process of such objects is highly labor intensive and time consuming. When we try to establish variability among the virtual objects, the traditional methods simply become impractical due to these limitations. For such situations, the solution lies in the procedural modeling, which can be defined as the creation of the visualization assets using computer algorithms based on a set of well-defined parameters. In this work, we parameterize wound geometry generation for the thigh model. The proposed model can generate arbitrary wounds on the base thigh object according to the parameters such as wound location, depth, size, and shape. This kind of variability in a training simulator will enhance the training experiences of the participants to a great extent. A surface flow model is also employed for blood flow visualization. Both the wound generation and blood flow simulation are implemented on graphics processing units (GPU) to achieve real-time performance.

Medical training simulators are important equipment for training novice health-care personnel. One application area of these simulators is training for wound debridement / treatment operations. Physical simulator kits are widely used for this purpose. These kits are large chests full of plastic wound (moulages) models and artificial blood packs (Figure 1). Although these kits serve the purpose, they are hard to carry around and most of the time they are for single-use. A more practical approach would be using computer simulation.

Medical Imaging 2012: Image-Guided Procedures, Robotic Interventions, and Modeling,
edited by David R. Holmes III, Kenneth H. Wong, Proc. of SPIE Vol. 8316, 831622
© 2012 SPIE · CCC code: 1605-7422/12/$18 · doi: 10.1117/12.911264

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 31 May 2023
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

Figure 1. Physical wound treatment simulator kits. These are often made of plastic and are for single-use [2].

In the work of Shen et al. [3], the debris on the contaminated wound was represented as a combination of texture images that were updated dynamically at different stages of the surgical debridement. Although they could capture the smooth transition between the dirty and clean states of the wound, the image based representation of the wound lacked the depth information, which is essential for immersive applications.

Lee et al. [4] followed a different approach to synthesize wound models on a 3D face model. In their approach, many wound images are used as samples to train a hue-based segmentation algorithm. After that, an input wound image is classified and its regions are mapped to depth layers, which is positioned on the face model according to the location selected by the user. Their approach does a better job in representing the volumetric nature of the wound model by using a disparity-based 3D geometry reconstruction algorithm. This model is intended to be used in computer animation, game, or virtual reality applications rather than a medical training simulation setting.

2. METHODOLOGY

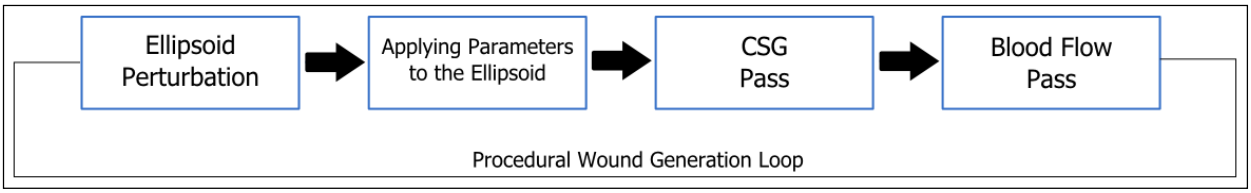

Figure 2. Structure of the procedural wound generation and surface blood flow model.

Constructive solid geometry (CSG) [5] is a solid modeling technique that applies Boolean operations to simple geometric primitives. In this work, we adopted a CSG based approach to generate the wound geometry. This process is composed of several steps as shown in Figure 2. The wound geometry is internally represented as an ellipsoid, whose vertices are perturbed with synthesized Perlin noise [6] by utilizing the GPU's vertex texture fetch functionality. Perlin noise is a procedural texture primitive that represents gradient noise and enhances the visual plausibility of the wound as it imitates the randomized pattern of the natural phenomena (Figure 3).

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 31 May 2023
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

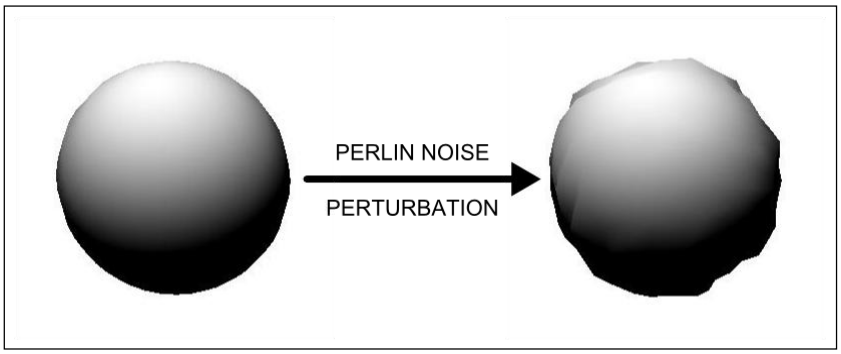

Figure 3. Perlin noise, which is a procedural texture primitive, is applied to the vertices of the sphere to obtain a natural randomized ellipsoid shape.

Perlin's noise algorithm consists of two main stages. The first stage is the offline one, in which pseudorandom values are generated for every position in 3D space. This part is implemented by utilizing hash functions that are used to resolve permutation and gradient texture primitives, which are generated using CPU runtime. The second stage of the algorithm is the online stage, in which the 3D position of the vertex that we want to perturb is used as an index to the generated permutation texture. Eight neighbor values are gathered from the permutation texture and the final "noise" value that will be used to alter the height of the vertex is obtained by interpolating between these eight values [6]. Perlin's noise algorithm therefore can be seen as a function that returns a floating point noise value for a given coordinate in 3D space. On top of this Perlin noise function, we applied spectral synthesis to obtain fractional Brownian motion (fBm). Fractional Brownian motion can be described as the result of Perlin noise being rescaled and added into itself. The fBm is implemented as

$$fbm(x,y,z) = \sum_{i=1}^{n} g^i . perlinnoise(l^i x, l^i y, l^i z) \quad (1)$$

Where g and l are the *gain* and *lacunarity* constants respectively. In our application, we chose *gain* to be 0.5, *lacunarity* to be 2, and number of *octaves* (n) to be 12. Each vertex of the wound model undergoes these operations on the graphics hardware and their locations are altered in a natural way.

The next step after obtaining a naturally perturbed ellipsoid is to apply constructive solid geometry (CSG) techniques to the ellipsoid model and the base model to which the wound is applied. In CSG, base primitives are combined by means of Boolean set operators and the series of operations are stored as binary trees. For our case, we used the difference operator that accepted the base thigh model and the ellipsoid model as operands (Figure 4). One problem with the CSG operations is that they require a lot of computational power if performed directly on the triangular meshes of the models. For interactive virtual reality applications, the traditional CSG implementation becomes impractical. On the other hand, screen-space CSG implementations [1] can perform the CSG operations and store the visible part of the resultant shape in the frame-buffer of the graphics hardware. Therefore in this work, we use an image based CSG implementation that utilizes the graphics buffers and handled entirely by the GPU to achieve real-time operation speeds.

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 31 May 2023
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

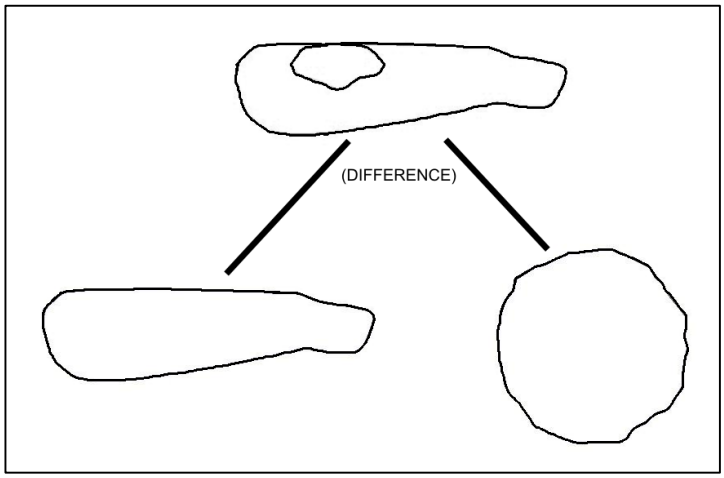

Figure 4. "Difference" operation is applied to the base thigh model and the generated wound geometry to obtain a wounded model.

Algorithm 1. Pseudo code for the screen space constructive solid geometry difference operation [1].

```
// We want to find B – A, graphics buffers used are:
// 1) Color buffer (CB) - what we see on the screen
// 2) Stencil buffer (SB) - used for complex masking operations
// 3) Depth buffer (DB) - stores the depth at each pixel
1 function ScreenSpaceCSGDifference(A, B):
// First draw the back faces of B that are inside A
2 Disable writing to CB, Clear SB, Enable DB
3 Draw back faces of A
4 Disable DB, Set SB to increment if depth test passes
5 Draw front faces of B
6 Set SB to decrement if depth test passes
7 Draw back faces of B
8 Disable SB, Enable writing to CB
9 Draw back faces of A
// Now, we need to draw the front faces of B to fix the depth buffer values
10 Enable DB
11 Draw front faces of B
// Draw B's front faces that are not inside A
12 Clear SB, Disable DB, Set SB to increment if depth test passes
13 Draw front faces of A
14 Set SB to decrement if depth test passes
15 Draw back faces of A
16 Disable SB, Enable writing to CB
17 Draw front faces of B
18 end ScreenSpaceDifference.
```

Texturing 3D models is another labor intensive step in traditional 3D modeling workflow and requires a great deal of experience to do it properly. As we have incorporated a parameterized wound geometry that does not have a fixed location or shape properties, static textures are not an option in this context. Fortunately, parametric representation of the geometry enables us to map the world coordinates of the wound geometry vertices to a 3D normalized coordinate system that can be used as volumetric texture coordinates. We have processed the 2D slices of the thigh region of the Visible Human Project [7] dataset and created a volumetric texture. This volumetric texture is dynamically accessed through the vertex shaders that are used to render the wound geometry (Figure 5).

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 31 May 2023
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

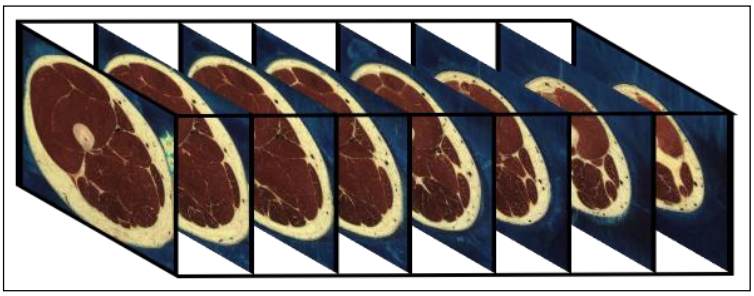

Figure 5. The texture slices that correspond to the base model are processed to obtain a 3D volume texture.

Although realistic blood flow models add a lot to the overall experience in terms of realism and level of immersion, they are often overlooked most probably due to their huge computational requirements. In medical training simulators, there are two main approaches for realizing blood flow: particle systems and surface flow models. In this work, we are employing a surface flow model that represents blood as a 2D height field texture that is continuously updated by the graphics hardware. The update procedure of the fluid height field is based on 2D velocity-advection processes. These physically-based fluid advection computations are all implemented on graphics hardware to enable real-time operation speeds. In order to obtain a plausible visualization of the effect, the height field texture undergoes several passes (Figure 8), and the final blood texture is projected on to the object by using the very same parameters that are used to generate the procedural wound geometry. In order to move the fluid in a physically plausible way, the surface flow model needs to have the knowledge of the topology. This knowledge is provided to the algorithm by means of normal and tangent maps, which are obtained and stored into graphics frame buffers by custom made pixel shaders. The acquisition of the normal and tangent maps is an important step of the surface flow model, because one can form a coordinate frame by using the tangent, normal, and the respective bi-normal at a particular location, which can later on be used to transform a given vector (such as the vector representing gravity) to this formed local coordinate system.

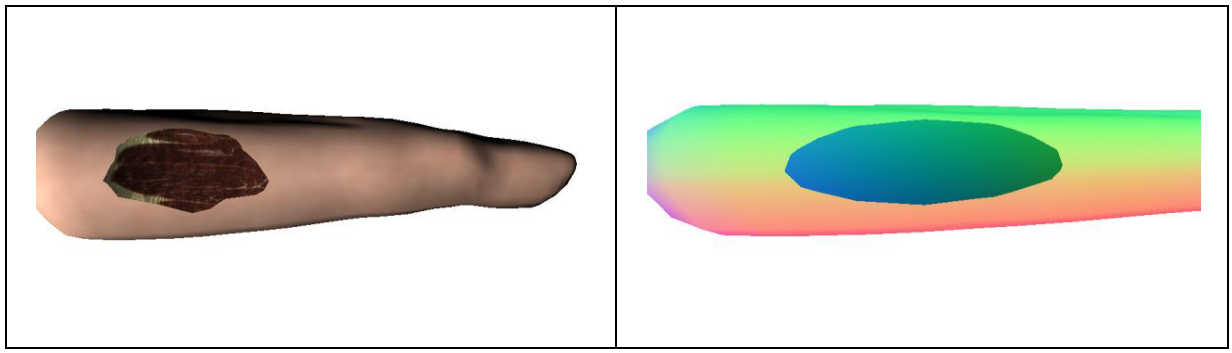

Figure 6. The thigh model along with the rendered wound and the corresponding normal map of the topology.

Therefore in the surface flow algorithm, a local coordinate frame is established at each pixel location and the gravity vector is transformed to these coordinate frames to obtain the overall flow pattern of a fluid over the given topology. This pattern is used as a rule map to advect and update the fluid volume on the topology.

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 31 May 2023
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

Algorithm 2. Pseudo code for updating the velocity component of the surface flow model.

```
1 function UpdateVelocity(acceleration, deltaT):
// Normal and tangent are obtained from the provided maps
2 binormal = cross(normal, tangent);
// Form a rotation matrix out of the vectors
3 rotation = [tangent , binormal, normal];
// Transform acceleration vector to tangent space
4 trAccel = rotation * acceleration;
// Update the surface velocity according to the transformed acceleration vector
5 newVelocity = oldVelocity + deltaT * trAccel;
// Also make sure that CFL condition is satisfied for a stable simulation.
6 return newVelocity;
7 end UpdateVelocity.
```

After the velocity component is updated, the fluid height field undergoes another pass, in which the new velocity values are used to move the fluid volume. The new velocity values are used to perform an inverse look-up on the height field to estimate the source of the fluid for the next simulation time step (Figure 7).

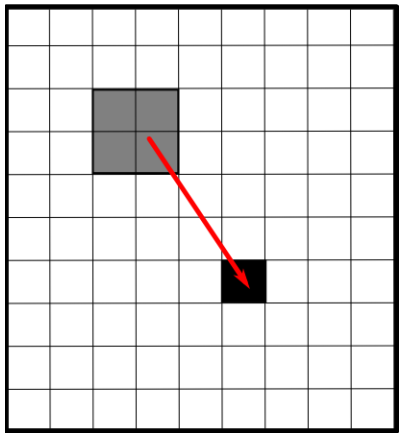

Figure 7. The updated velocity values are used to perform an inverse look-up for the current cell (black). The output of the look-up is a neighborhood of cells (gray) whose fluid volumes are advected to the current cell.

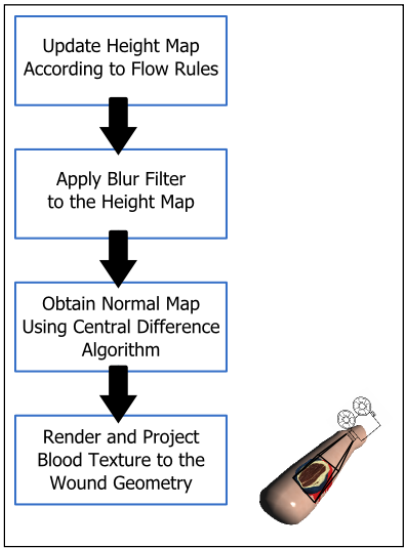

Figure 8. The flow chart for updating the blood texture.

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 31 May 2023
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

3. RESULTS

Wound models are important elements of wound treatment training procedures. Procedural generation of wound models in a simulation is an important enhancement over the static models as it allows for varying scenarios and test cases. This capability of creating variations is accomplished by the parametric representation of the wound. Currently, our procedural technique supports parameters such as location, orientation, depth, and size. By using different combinations of these parameters, it is possible to represent multiple situations during the simulation (Figure 9).

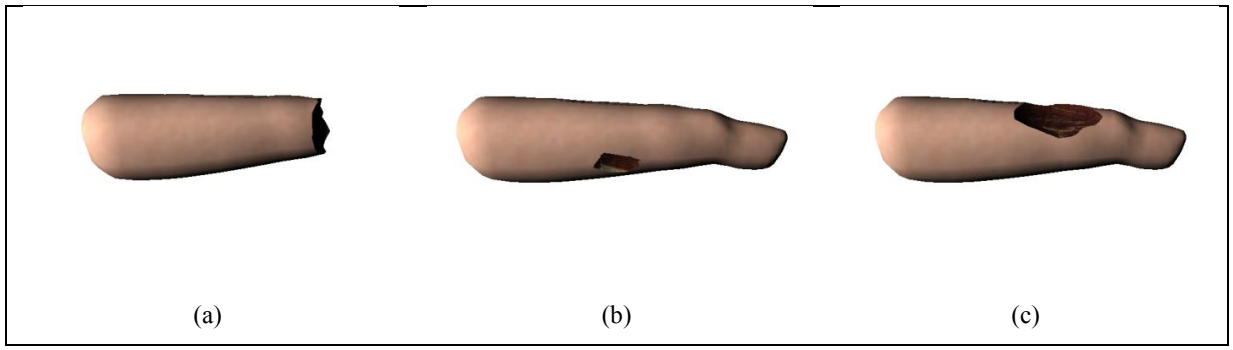

Figure 9. By varying the parameters -location (a), depth (b), and orientation (c)- supported by our procedural model it is possible to obtain various simulation cases.

Although there have been previous attempts for generating wounds procedurally, to our knowledge this work is the first one to generate 3D wound geometry that uses volumetric textures and surface blood flow animation to increase the realism of the simulation and immersion of the participants. Another important aspect of this work is that it harnesses the computational power of graphics hardware, which makes this model readily embeddable to a previous work without affecting the system's overall performance. As a showcase for our proposed framework, we used our model in a surgery scene for a thigh model (Figure 11).

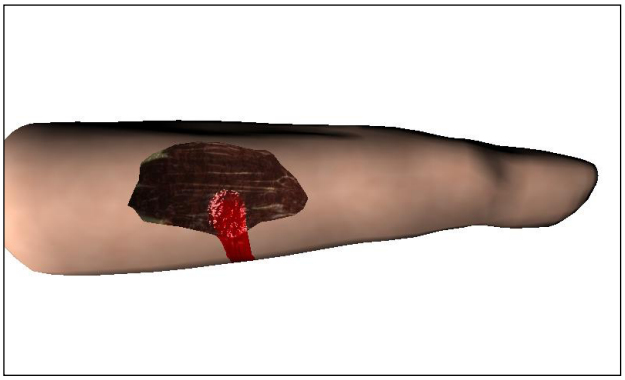

Figure 10. Wound geometry is generated on the thigh model along with the flowing blood on the surface of the wound and thigh model.

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 31 May 2023
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

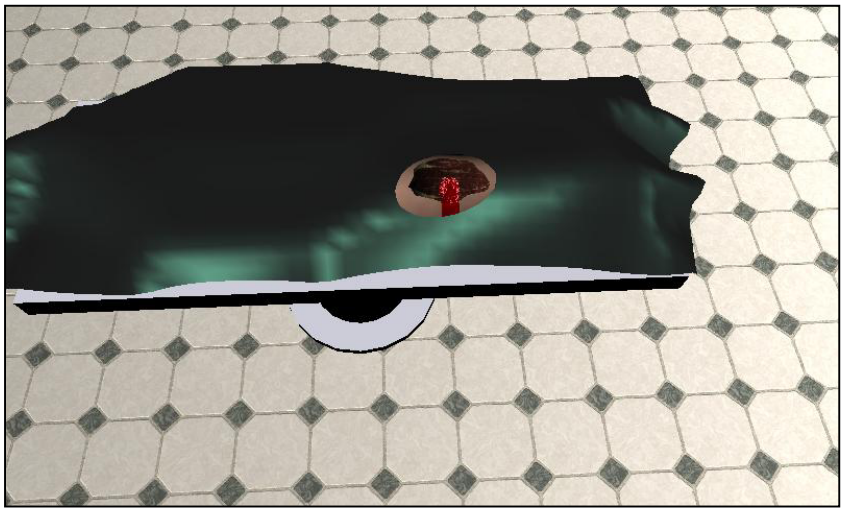

Figure 11. The proposed procedural techniques are used in a surgical simulation environment.

4. CONCLUSIONS AND FUTURE WORK

In this work, we defined a procedural framework to generate and visualize 3D wound geometry for an arbitrary body model. Procedural nature of the system allows repeatability of the simulation through the usage of the same parameters. Another advantage of the procedural modeling in this context is that training scenarios can be varied with different models without the need of labor intensive and expensive 3D remodeling. Wound geometry creation process is realized as a constructive solid geometry (CSG) operation, in which a perturbed ellipsoid model is subtracted from a base model to obtain a wounded body part. In addition to the wound geometry creation, we also implemented a physically-based surface flow model for realistic blood flow simulation. Both the wound geometry creation process and the surface blood flow model are implemented on graphics hardware, which makes our approach to be readily embeddable to other simulations without loss of performance or generality.

Our approach has plenty of room for improvement. Currently a wound is defined by parameters like location, depth, and size. One feature that we want to add to this system is higher-level parameter definitions. The medicine literature has classified wounds into 6 types: abrasion, laceration, avulsion, incision, puncture, and amputation (Figure 12). We will define these wound types as a parameter set for our system.

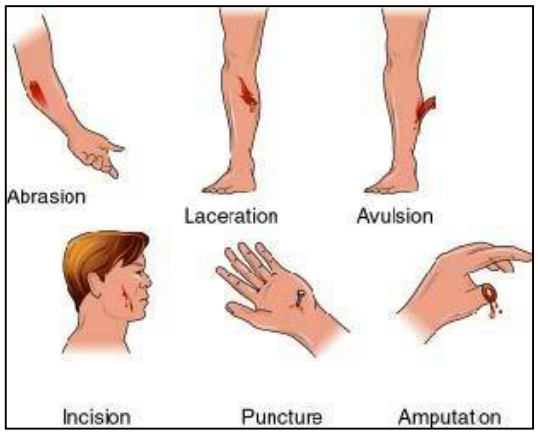

Figure 12. Wound types defined by the medicine literature [8, 9].

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 31 May 2023
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use

REFERENCES

[1] F. Kirsch and J. Döllner, "OpenCSG: a library for image-based CSG rendering," 2005, pp. 49-49.

[2] Buyamag Inc. (1996). Wound Trauma Military Simulator. Available: http://www.buyamag.com/wound_suturebandaging.php

[3] Y. Shen, J. Seevinck, and E. Baydogan, "Realistic irrigation visualization in a surgical wound debridement simulator," Studies in Health Technology and Informatics, vol. 119, p. 512, 2005.

[4] C. Y. Lee, S. Lee, and S. Chin, "Multi-layer structural wound synthesis on 3D face," Computer Animation and Virtual Worlds, 2011.

[5] D. H. Laidlaw, W. B. Trumbore, and J. F. Hughes, "Constructive solid geometry for polyhedral objects," 1986, pp. 161-170.

[6] K. Perlin, "Implementing improved perlin noise," GPU Gems, pp. 73-85, 2004.

[7] V. M. Spitzer and D. G. Whitlock, "The Visible Human Dataset: the anatomical platform for human simulation," The anatomical record, vol. 253, pp. 49-57, 1998.

[8] L. Snyder, "Wound basics: types, treatment, and care," RN, vol. 71, pp. 32-37, 2008.

[9] Nursingcrib. (2008). Types & Causes of Open Wounds. Available: http://nursingcrib.com/nursing-notes-reviewer/types-causes-of-open-wounds/

Downloaded From: https://www.spiedigitallibrary.org/conference-proceedings-of-spie on 31 May 2023
Terms of Use: https://www.spiedigitallibrary.org/terms-of-use