

STATE UNIVERSITY OF NEW YORK AT BUFFALO
Mechanical and Aerospace Engineering Department.

**MAE 574 Virtual Reality Applications &
Research**

FINAL PROJECT

“Tissue Cutting & Bleeding Effect Simulation”

NAME: Regeesh Britto & Leng-Feng Lee

DATE: 4 MAY 2004.

Student Name: Regeesh Britto
Student ID: 3041-2424
UserName: rjbritto@

Student Name: Leng-Feng Lee
Student ID: 2963-2068
UserName: llee3@

This course was enrolled in Spring 2003 semester.

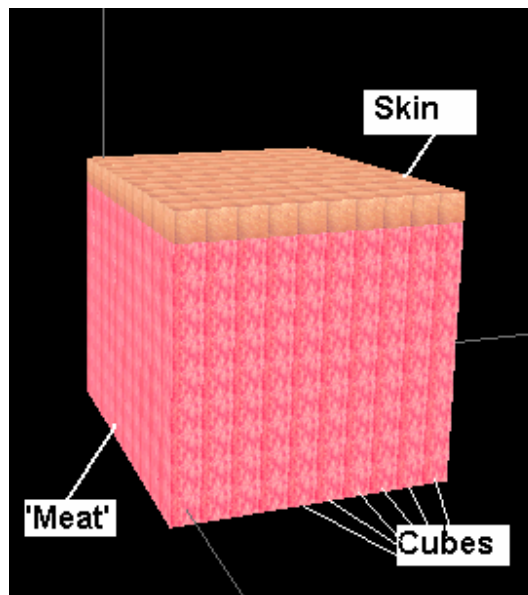
Project Title: Simulation of Tissues Cutting & Bleeding Effects.

Introduction:

This project uses Microsoft VC++ and the GLUT library to simulate the Tissue Cutting process. To make the cutting process more realistic, we created: bleeding effect on the tissue; blood stains on the needle that performs the cutting; and implement of a fountain that spilt blood on the first cutting position. To make the simulation more realistic, texture mapping was also added.

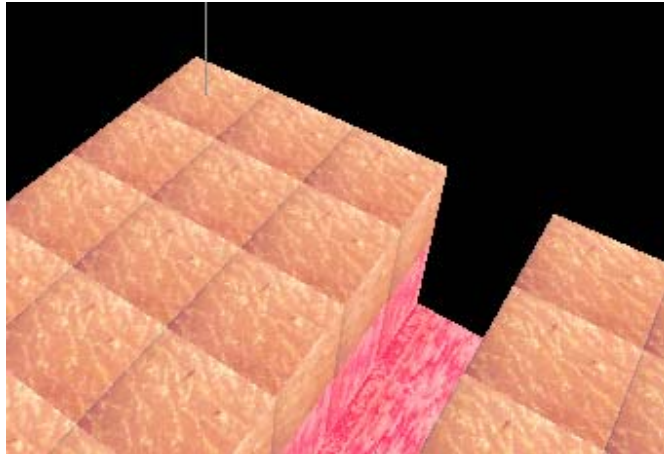
Modeling Concepts:

Tissue Modeling:



The complete tissue model.

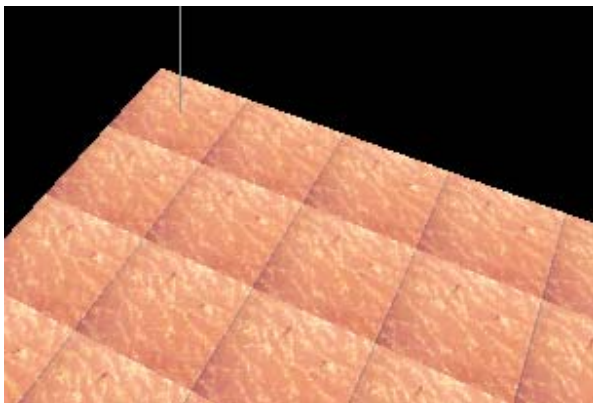
The tissue is modeled using stacks of small cubes. When a cut is detected, the corresponding cube is not drawn. We can increase the numbers of cubes, and decrease the size of the cube to increase the resolution of the cutting process.



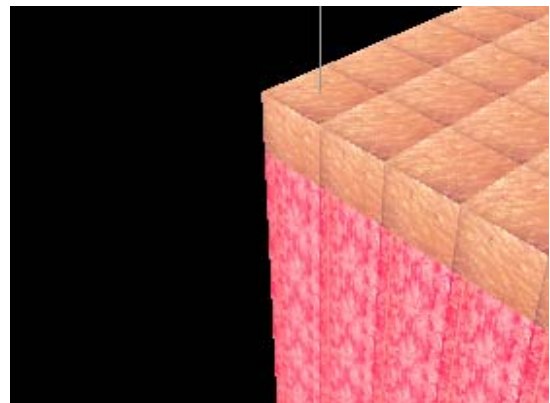
Cut is detected and no cube was drawn.

Texture mapping, was not possible with `glutSolidCone()`, which we had planned to use at first, so a custom function had to be built to draw each face of the cube, `Solid_Cube()`. But what we found out was that as the number of cubes increased the efficiency of the program reduced. This was because the program had to draw huge number of faces and also texture them. Therefore we introduced back face culling so that the face that was not visible was not drawn or textured. The GLUT function used was `glEnable(GL_CULL_FACE)`

Texture mapping was used to separate the “skin” and the “meat” with the top layer of cubes made the skin and the rest flesh. These are shown as follow:



The “Skin”



The “Meat”

The tip of the needle was used to detect whether a cube had to be blanked. If the cube was somewhere below the top layer, then all the cubes above it in the y-direction was also blanked. This was done for a more realistic deep cut effect.

Functions used to generate Tissue: `Solid_Cube(float, float, float, float, float)`, `drawcube()`, `drawblank(blank)`, and `checkblank(blank)`.

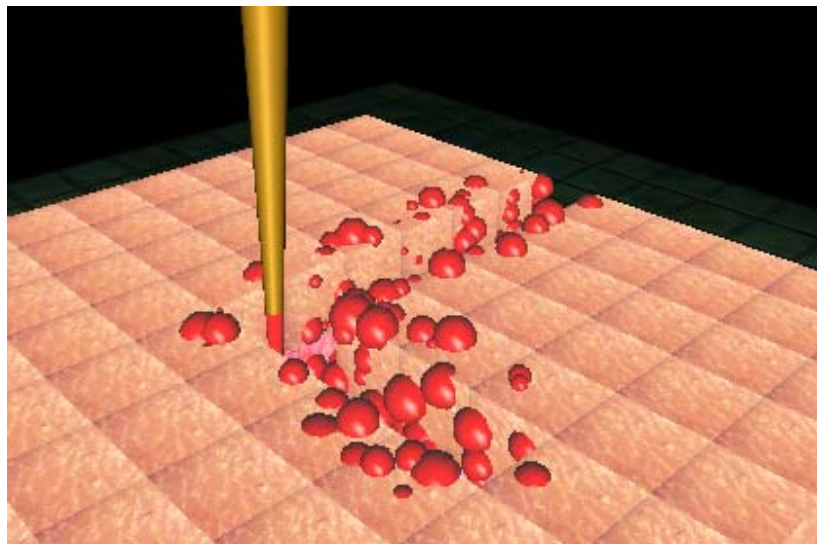
-blank is a struct to save the i, j, & k index of the cube.

Bleeding Effect Modeling:

The bleeding effect is activated once a cut is detected. It generates bloods on the surrounding four side of the blank cube. Initially, all the cubes has a unique ID, which is the $[i, j, k]$ of their locations. We assign ID of 1 to empty cube and ID of 0 to not empty cube. Each cube will change its ID to 0 if the needle is within that cube.

	Cube[i][j-1][k]	
Cube[i-1][j][k]	Cube[i][j][k]	Cube[i+1][j][k]
	Cube[i][j+1][k]	

When the needle is within a cube, a specific Blood Pattern ID will also be assigned to the cube. This Blood Pattern ID will be used to generate 20 random (random in x, y, z position and also radius) “blood drop” at the empty cube surrounding (4 Sides). The use of this BloodID with the checking of Empty/Filled cube of its surrounding cubes, we can determine when we should draw the blood. That is, for example, if Cube[i][j][k] has a ID of 1, we generates the Bloods at its surrounding, this blood pattern is assigned a BloodID to remember the randomly generated blood pattern. It then check its surrounding cubes, for example, if Cube[i-1][j][k] is 0, we will draw the blood on it.

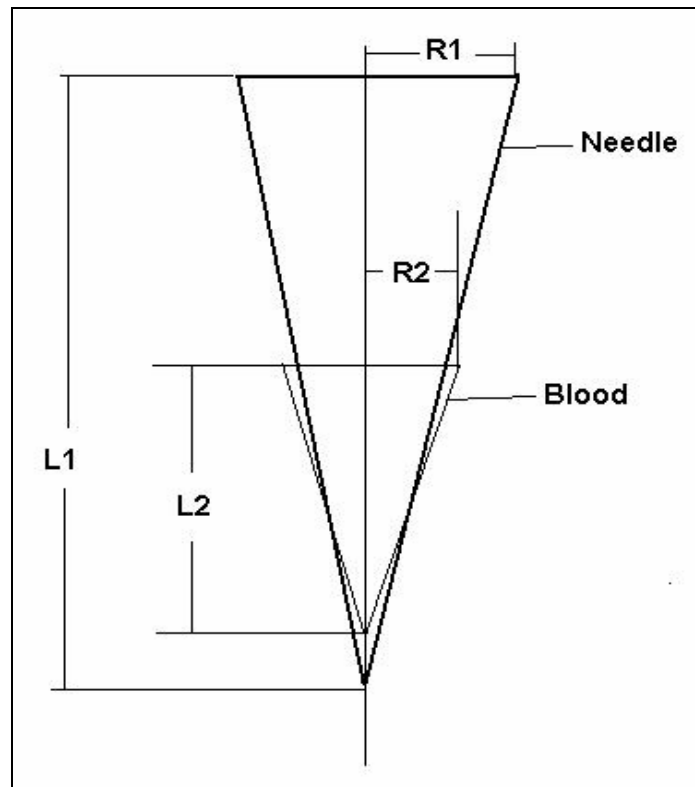


Randomly generated blood pattern drawn.

Functions Used: CreateBlood(int), & DrawBlood().

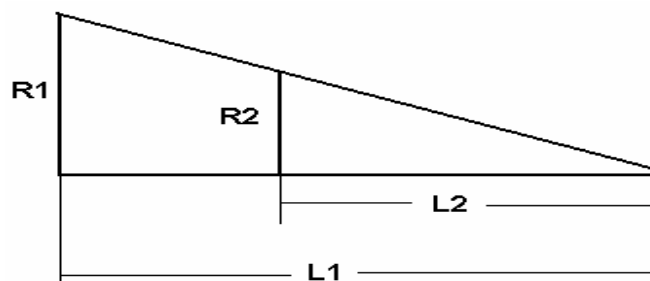
Blood Stain on Needle Effect:

Since there is bleeding when cutting the tissue, it is nice to make some blood stains on the needle. The needle is drawn using `glutsolidcone()`. We fake the effect of stain by create another cone at the outer layer of the needle cone. As illustrated in the following picture:



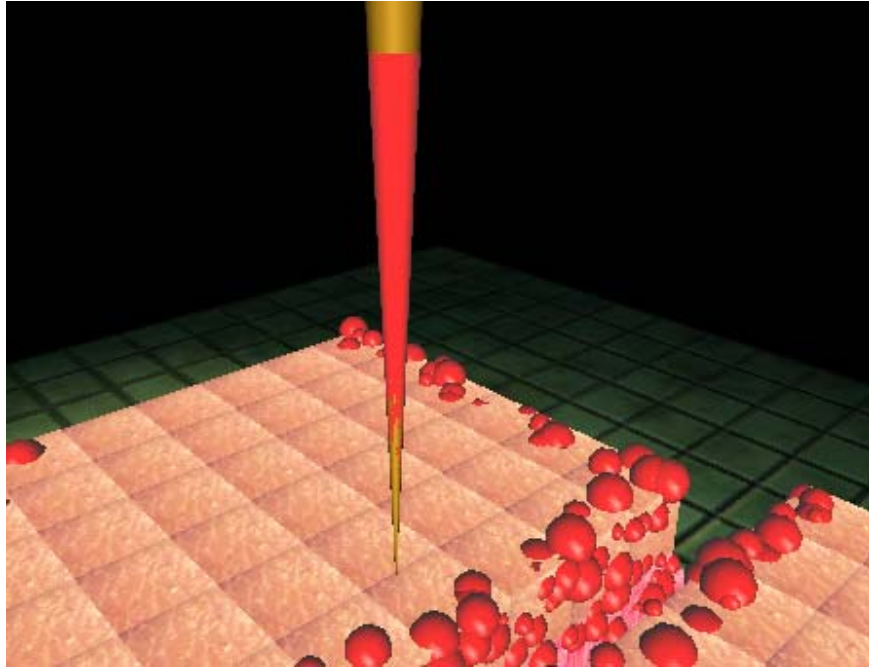
Needle Blood Effect.

To use `glutsolidcone()`, we need to give the radius of the “blood cone”. We can determine the depth of cut $L2$ by recording the deepest cut it has reached before. Then the following equation is used to calculate $R2$:



$$R2 = \frac{L2}{L1} R1$$

We update the value of $L2$ from the position of the needle, and calculate $R2$ value. When the BloodCone is drawn, we translate it a bit higher and make the radius a bit larger, create an effect of blood stain. As shown in the following:

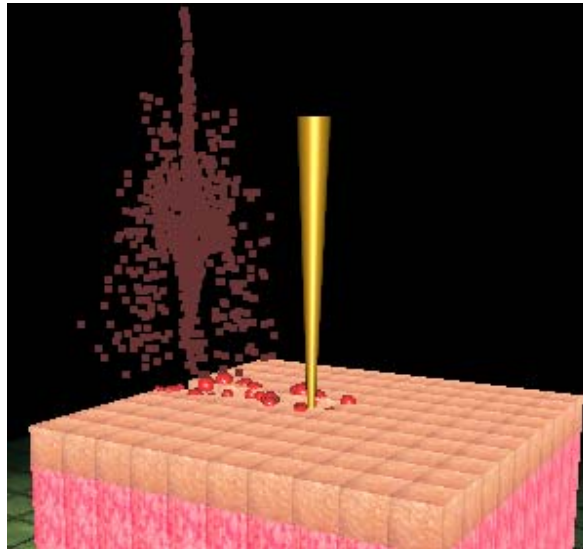


Blood stain effect on the needle after a deep cut is performed.

Functions Used: glutsolidcone()

Blood Fountain Effect:

We adapted the code that was created by Philipp Crocoll (www.codecolony.de). A blood jet was activated at the very first cut cube to simulate a blood spurt at the point of incision. Since the particle generation engine required a lot of computing power to run especially in such highly interactive program it was not included as a default feature.

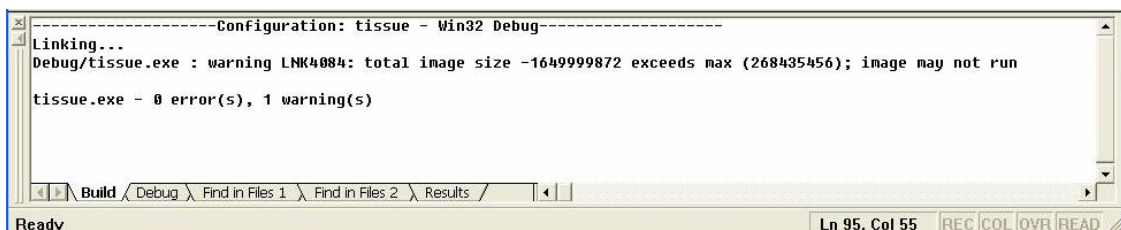


Blood Fountain Effect.

Limitations:

This program requires very large amount of computing power because of the particle generation engine, blood effect and the texturing. One of the biggest limitation encountered was that we couldn't increase the number of cubes beyond 20 in the xyz direction. Further analysis of the problem on the net showed that Windows has a limit for the total size of static code and data of 256MB. If the amount of static code and data exceeds this size, the image may not execute. This is a limitation in the operating system and not in VisualC++ or its tools. A detailed explanation can be obtained at

<http://h18009.www1.hp.com/fortran/kb/q1019.html>

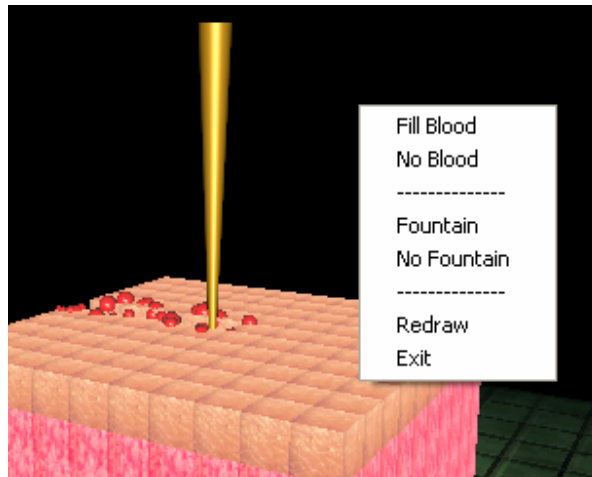


Some User Friendly Options:

KeyBoard: we used the keyboard as the main input option for this simulation.

Mouse: Mouse is used to control the movement of the needle.

Menu: The Menu options is shown below:



Help file: we also create a “Help file” that can be access by mouse click.

We include our code in the CD:

1. tissue.cpp
2. fountain.h
3. textures.h
4. textures.cpp
5. general.h
6. VR_instruction.txt
7. glut.h
8. basetsd.h
9. Executable .exe file is also included.
10. WORD document and PDF of this report is also included.