POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

EXECUTIVE SUMMARY OF THE THESIS

Ultrasound Simulation with Deformable Mesh Model from a Voxel-based Dataset

TESI DI LAUREA MAGISTRALE IN
COMPUTER SCIENCE AND ENGINEERING – INGEGNERIA INFORMATICA

AUTHOR: NICOLO' STAGNOLI

ADVISOR: MARCO GRIBAUDO

ACADEMIC YEAR: 2022-2023

1. Introduction

Ultrasound is a widely used, low-cost imaging modality which plays an important role in clinical diagnosis. Today, technology has developed a lot and it is possible to use computer simulations to train in the medical field. The aim of this work is to explore volume rendering techniques and develop a three-dimensional representation of the human body so that it contains also "internal" texture sampled from the AustinMan dataset. This internal texture is then used to build new images from the intersection of a plane with the body. The power of this concept lies in the fact that is possible to build slices of the body in any orientation, not just horizontal. The body can also be set in different poses and the position of the internal points is recalculated.

Austin Man Dataset

AustinMan is a voxel model of the human body that is being developed for simulations. This dataset was developed from a real MRI scan by segmenting the color cross-sectional (transverse plane) anatomical images. It contains 1878 horizontal (only) slices of a whole human body in different resolutions. The grayscale value of each pixel corresponds to a different layer of the body, such as tissues, bones, and organs, for a total of 64 layers.

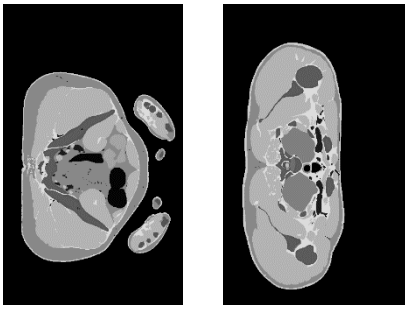

Figure 1: Sample images from the AustinMan dataset.

Ultrasound Simulator

Overall, the main advantage of ultrasound simulators is that they provide new, innovative ways for learners to build up mental models. Yet, the most important feature of an ultrasound simulator is to provide feedback, which conforms to the concept of a mental model that must be updated. Unfortunately, there are some disadvantages associated with these simulation systems. First, these systems are now very expensive, and it is not possible to provide one for every student to train. These systems are generally quite bulky and require a very high computational power, which leads to being forced to use them in places set up for this purpose. Furthermore, many of these systems consist in a "static" simulation, meaning that the models involved in the simulation are fixed and cannot be deformed. These models are often hand-crafted. A lot of effort is made to produce and validate these models, increasing the total cost of the work.

Purpose and Motivation

The solution chosen to overcome the disadvantages described above is to provide a reliable simulation of an ultrasound activity, using an ultrasound probe to perform the examination. This simulation works in a lightweight environment that can potentially run also on portable devices. All the three-dimensional models of the human body skin and its internal tissues and organs involved in this simulation are generated with voxelization algorithms. These voxelization algorithms in general perform triangulations of surfaces by sampling values from a discrete field of three-dimensional points. When the ultrasound probe intersects the body, the "internal" texture of it is shown on the screen in the scene.

Mesh Model Deformation

The main point of the developed simulation is the possibility to deform the human body to set it in a general pose. Arms and legs can be oriented in any direction. Doing this, a technique to map every part of the body to the "internal" texture, and deform it, accordingly, is required.

2. State of the art in Ultrasound Simulation

In achieving educational outcomes, computer-based simulators mimic the ultrasound image produced within a computer. The methods to simulate ultrasound images can be categorized into:

- Interpolative
- Generative image-based
- Generative model-based

The interpolative approach uses prerecorded three-dimensional ultrasound volumes and slicing techniques, that can be combined with postprocessing like deformations and artificial shadow insertion in the final image.

Generative image-based models rely on Machine Learning and Deep learning techniques, Particularly, almost all the approaches to realistically simulate ultrasound images in this way are based on generative adversarial networks (GANs).

The generative approach simulates ultrasound images using geometry from imaging systems like computed tomography (CT) and magnetic resonance (MRI), or it is based on mesh models. Generative model-based ultrasound simulators create an image by extracting a bi-dimensional slice from the model and texturizing it.

Objective of this work

In the past year, here at Politecnico di Milano, Diego Zucca presented a lightweight reliable simulation of an ultrasound activity, using an ultrasound scanner and an ultrasound probe to perform the examination, running on VR (Virtual Reality headset). The aim of this work is to enhance the previous work by Diego Zucca to account for model deformation. Setting the model in different poses could be interesting to see what happens to internal body part when the pose is changed, or to see what the effect of involuntary body movements are, like breath, on the internal texture. For this reason, a technique to map every part of the body to the "internal" texture, and deform it, accordingly, will be investigated.

This work led to the realization of a hybrid method for ultrasound simulation. Since images on the screen are built by pre-recorded images, from the AustinMan dataset, it is an Interpolative method. The concept of mesh model deformation, and its application to the interpolated image, however, derives from a three-dimensional model. We can categorize this method as an Interpolative Model-Based.

3. Voxelization Algorithms

Simple Voxelization

The simplest method to construct a three-dimensional model from the stack of images of the dataset is to instantiate a cube for each colored pixel, since black color corresponds to air. However, due to the high number of pixels in the dataset, this simple approach is not feasible because memory overload. The number of vertices is too high, and most of them are inside the body, so not even rendered.

The idea is to algorithmically create vertices and triangulate them to make simple cubes. Each pixel in the dataset images corresponds to a cube. If one face of a cube being drawn is covered by another cube, the face is inside the mesh, and it doesn't need to be rendered. Doing so, only the external part, the skin, of the human body is drawn, while the inside is empty.

This simple approach is also used to build each layer of the human body. Different meshes are built iterating each time on the whole dataset, taking only pixels with a specific value of the grayscale value.

Marching Cubes

The Marching Cubes algorithm is used to approximate an isosurface by subdividing a region of space into a three-dimensional array of rectangular cells, which is the most popular method for isosurface rendering. It is mainly used in medical applications as indirect volume rendering technique and in video games to procedurally generate terrains (meshes, in general) from a discrete field of values, which can be randomly generated from noise or taken by image maps.

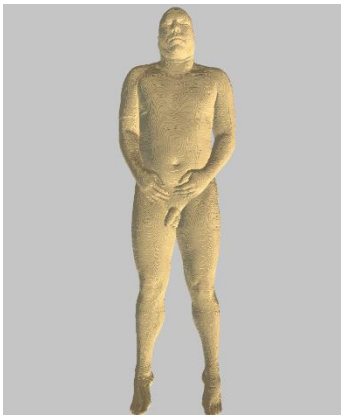

Figure 3: The mesh generated with the Marching Cubes Algorithm.

The basic idea of Marching Cubes is that voxel could be defined by the pixel values at the eight corners of the cube. If one or more pixels of a cube have values less than the user specified isovalue, and one or more have values greater than this value, we know the voxel must contribute some component of the isosurface. By determining which edges of the cube are intersected by the isosurface, we can create triangular patches which divide the cube between regions within the isosurface and regions outside. By connecting the patches from all cubes on the isosurface boundary, we get a surface representation.

4. Ultrasound Simulation with Deformable Mesh model

The possibility of being able to perform ultrasound scans in distinct parts of the patient's body is one of the objectives of this work.

3D Texture and UVW Mapping

A 3D texture is a bitmap image that contains information in three dimensions rather than the standard two. 3D textures are commonly used to simulate volumetric effects such as fog or smoke or to approximate a volumetric 3D mesh. In this work, a 3D texture is made by building a three-dimensional matrix from the dataset images. Each horizontal slice of the matrix corresponds to a slice image in the dataset.

3D texture can be rendered just like normal 2D textures, with the same types of filtering and interpolations to calculate final pixel values; the only difference is, obviously, the presence of one extra dimension for texture coordinates: there will be UVW coordinates. 3D texture can be applied to any mesh.

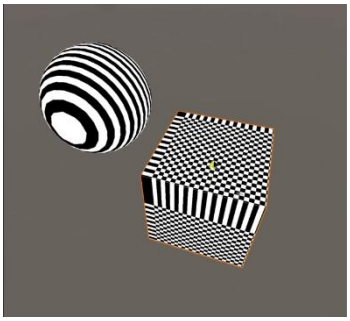

Figure 2: Example of 3D textures applied to a cube and a sphere.

The two main components of which it is composed are the Vertex Shader and the Fragment Shader. In

the Vertex Shader the position of the object is transferred from object space to the camera's clip space in homogeneous coordinates. The most important part takes place in the Fragment Shader, where the UVW values are used as coordinates to indicate a section of the 3D texture from which extract the color to be assigned to the material.
The idea is to exploit the possibility to use the position of the vertex of the invisible plane of the probe as UVW coordinates during the sampling phase of the 3D texture. The vertex shader step is exploited to make use of the rendering pipeline in a clever way. Each of the plane vertices is assigned with specific UVW coordinates. Every fragment within these vertices will be automatically sampled and interpolated by Unity's built in rendering pipeline. In this way, there is no need to implement specific interpolation algorithms and filtering of the textures.

Ultrasound Probe and Screen

In this work, to simulate the ultrasound probe, a plane is used. This plane is set in the world origin. Every time the plane position or rotation changes, the UVW coordinates of each vertex of the plane are set to the world position coordinates of the vertex itself. By moving the plane in the space, we obtain a static mapping of the 3D texture.
The screen is the same plane used for the probe, the same UVW coordinates are applied to vertices of the screen.
The plane mesh is realized in Blender, by creating a simple plane mesh and subdividing it until reaches an acceptable "resolution". In this way, the plane is made by a higher number of vertices, not just 4 as in standard plane meshes. The idea is to exploit this higher "resolution" of the plane to have the possibility to set cluster of vertices mapped to a specific part of the 3D texture dynamically.
This concept is essential to realize the UVW mapping with the deformable mesh model. Differently from the static case, different parts of the 3D texture need to be rendered on the plane, and these parts in general are no longer aligned (in the UVW space), due to the possibility of deforming the model rig. This mapping would not be possible if the plane had only 4 vertices.

Mesh Model Deformation

One of the main points of this work is to make the human model deformable, (just rotation and translation of body parts), with the mesh carrying the "internal" texture along the way. To do this, two approaches are presented.

The first approach consists in manually placing spheres along arms and legs. These spheres are attached to the model rig. The initial position of these spheres is stored at the start. The idea is that, whenever the model rig is deformed, each point position of the cutting plane is compared with all the spheres: if the point is inside one of them, the inverse of the sphere rotation and translation is applied to the UVW coordinates of that vertex. If the involved sphere is still in the original position, there is no rotation and translation, and the result is the same of the "static" version.

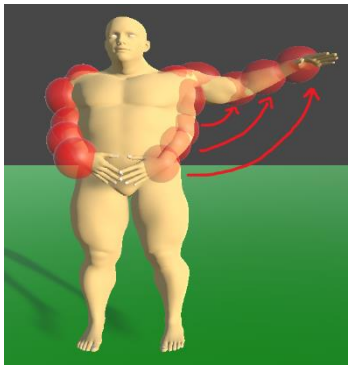

Figure 4: The approach with Sphere Colliders: Rotate and Translate.

This method works, but results applied to this precise human model is not satisfying, due to the nonstandard base position of the mesh (so also of the 3D texture), the initial overlapping of some body parts, and the high inaccuracy of the spheres to approximate body parts. The approach, however, is working.
To improve the accuracy of collision detection with colliders, the idea is to exploit the human model geometry instead of the spheres to make more accurate colliders. However, Unity's built-in Mesh Collider component can only create convex colliders for the whole mesh, so this is not useful. The final approach consists in automatically generating small convex mesh colliders for each part of the body to better approximate its bounding. Exploiting the bone weights of the human model rig, the model is subdivided into sub-meshes by considering which vertex is attached to which bone. These sub-meshes colliders are regenerated every time the model rig is deformed. As in the previous method, the initial position and orientation of each collider is stored at the start to subsequently apply the inverse transform to plane points.

Breath Simulation

To simulate the breathing movement of the body, the chest bone is animated. For a period of about 4 seconds, its scale increases, and decreases. Applying scale to a bone result in scaling all the vertices assigned to that bone. By applying scale deformation to a bone, the same scale is applied also to all the child bones, so also the head and the arms are enlarged. To prevent this, the same animation, but with inverse scale factor, is applied

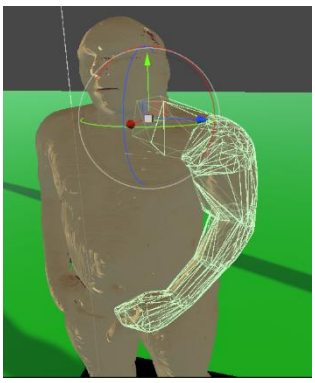

Figure 6: Colliders of the left arm generated with the bone weights.

to all the child bones that doesn't need to be enlarged. Colliders are automatically recalculated in each frame. The local scale of each bone/collider is also used to calculate the inverse transformation to be applied to plane points.

5. Conclusions

The proposed solution consists of an application where the user can freely move in the scene and interact with the environment. The body mesh generated with Marching Cubes algorithm from the AustinMan dataset is placed in the center, and the screen for visualizing the ultrasound image is placed behind. The produced imaged is colorized with a pre-defined color for each body layer. This is because the grayscale version of the image can create confusion if the user assumes that dark areas are the densest, which is not the case since the dataset color encoding didn't follow this assumption. The probe plane position and orientation can be set by the user. Layers of the body, such as skin, muscles, bones, organs, can be set to be visible or not and be closely inspected. Body parts can be selected to set the orientation of the corresponding bone as the user wishes. There is also the possibility to simulate the breathing movement of the body, and its effect can be seen on the screen.

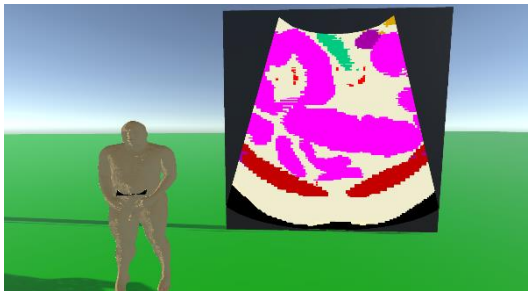

Figure 5. The Developed Ultrasound Ecography application.

Comparison

In comparison with the previous work, there are significant improvements.

The first is about the model mesh geometry. In the previous work, a third-party model was manually set in position to be in some way aligned with the 3D texture. This model, however, could not fit fully and exactly the texture because its shape didn't correspond to it. Its function was only to be like a placeholder. In this work, the model mesh shape used to represent the human body corresponds perfectly with the shape of the 3D texture, because it is generated from it. This is also essential to make the collider subdivision, that is needed to map points when the model is deformed. Another improvement consists, indeed, in the possibility to change the body position to see what happens to the internal texture when the body is deformed. This feature was not present in the previous work. Last, the proposed solution contains a full body scan, while in the previous work only some parts were available. Also, there is no more need for model registration: the mesh model is already placed in position and aligned with the 3D texture. By simply placing the probe in the desired position we get a scan of the corresponding body part. In the previous work, to do this, a specific configuration scene called "Developer Mode" was set to align the body mesh model with the desired part of the texture.

Critical Points

Despite the proposed solution is working well, there are still some criticalities about some cases in which the approach is not working correctly, showing a distorted or segmented images on the screen.

The most important critical point is about the texture sampling derived from the model mesh deformation. When the body is in the standard pose, the 3D texture is sampled perfectly.

However, when the body pose is changed, there could be some regions in which the 3D texture is not sampled correctly. These critical regions arise in body parts corresponding to colliders boundaries. Colliders are automatically generated by considering the bone weights. Bone weights are also generated automatically, but the weighting strongly depends on the model rig, which is in this case done manually. Increasing the number of bones of the rig helped to reduce the size of these problematic regions, but in some cases, despite being smaller, these regions increased in number.
Another important problem that arose during the development of this work was the illness about the position of the human body in the dataset images, which came from a real MRI analysis. This led to difficulties during the rigging of the model because of overlapping geometry between hands and body, and because automatic bone-weight tools work much better when the base pose of the model is the T-pose. The previous criticality about collider boundaries could depend on this also.

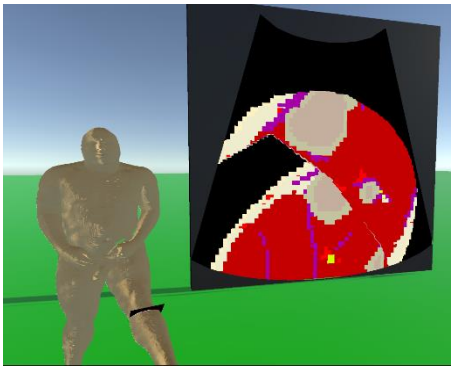

Figure 7: An example of wrong re-mapping of the texture after model deformation

Further Research

This work can be enhanced and improved in more ways.
First, a better subdivision of the mesh, rig, and colliders could be done to improve the inaccuracy encountered when scanning near colliders boundaries.
Also, a faster version of the proposed algorithms could be implemented. Now, the model mesh deformation and calculation of the UVW coordinates is entirely made with the CPU, with the GPU working only on fragment color interpolation. Research could be made to find if it is possible to exploit the GPU to directly apply the model deformation to every fragment UVW coordinates, exploiting the fragment shader programmability.

Ray-Casting

This work led to an Interpolative Model-Based ultrasound simulation application. It would be interesting to apply the model deformation technique with the Direct Volume Rendering (thus, Generative model-based approach), already introduced Ray-Casting technique. Rays could be cast in the common way, passing through the image plane. The same mesh models generated from the AustinMan dataset can be used for collision detection and model deformation. When a ray intersects the mesh, the ray origin can be recalculated as if it was intersecting the same collider but in the base pose, to simulate the ray pass through the correct portion of the volume, when the model pose is deformed.

Bibliography

D. Zucca, "Low-cost Virtual Reality Simulation of Medical Ultrasound," Politecnico di Milano, 2022.

Sascha Bettinghausen, "Real-Time GPU-Based Ultrasound Simulation Using Deformable Mesh ModelsBenny Bürger", *IEEE TRANSACTIONS ON MEDICAL IMAGING*, vol. 32, March 2013.

W. E. Lorensen, Harvey E.Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," in *SIG '87*.

J. W. Massey and A. E. Yilmaz, "AustinMan and AustinWoman: High-fidelity, anatomical voxel models developed from the VHP color images," in *Proc. 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEEE EMBC*, Orlando, 2016.

6. Acknowledgements

Here you might want to acknowledge someone.