Requirements for VR Drilling Simulator

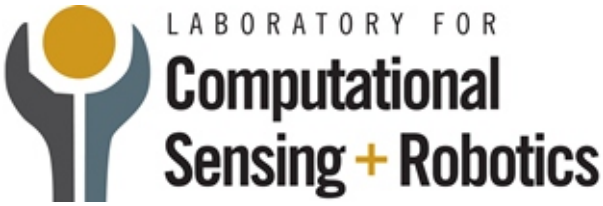
A FUNCTIONAL REQUIREMENTS DOCUMENT

submitted by

KESAVAN VENKATESH
JONATHAN WANG
YI WANG

Virtual Reality Drilling Simulator for Laminectomy (Team 19)

Project Mentors: David Usevitch, Hisashi Ishida, Adnan Munawar



Laboratory for Computational Sensing and Robotics
Computer Integrated Surgery II
601.496/656
April 2023

Introduction

We developed a color-guided virtual reality (VR) drilling simulator for training surgeons on laminectomy operations. This overlay visualizes critical structures (e.g., dura, spinal cord) and relative distances from them (color gradient indicates increasing distance from the anatomy in the coordinates of the simulator), both of which are not available intraoperatively. We conducted a user-study to evaluate the differential impact this color-guidance had on drilling accuracy and operation time when training with the simulator.

Purpose of this Document

The Functional Specification Document (FSD) provides detailed information on how the system solution will function and the requested behavior. This document will connect the high-level requirements identified by surgeons to the technical details of the software implementation. Included in this document will be use cases, system inputs and outputs, process flows, diagrams, and mock ups.

Project Scope

This simulator is intended to be used as a training tool for practicing orthopedic surgeons. The current anatomy and color-overlay are specific to laminectomy operations, but the system is modular enough to be adapted to other drilling procedures in the future (e.g., mastoid drilling). Furthermore, this project focused solely on visual feedback; future iterations may look to improve audio/haptic feedback.

Terms/Acronyms and Definitions

Term/Acronym	*Definition*	*Description*
VR	Virtual Reality	Computer-generated simulation of 3D environment (in this case, lumbar spine, drill, and OR lighting) that is immersive and physically representative
SDF	Signed-distance field	3D field where each voxel is assigned a float (+/-) that indicates closest distance to the anatomy of interest

Risks and Assumptions

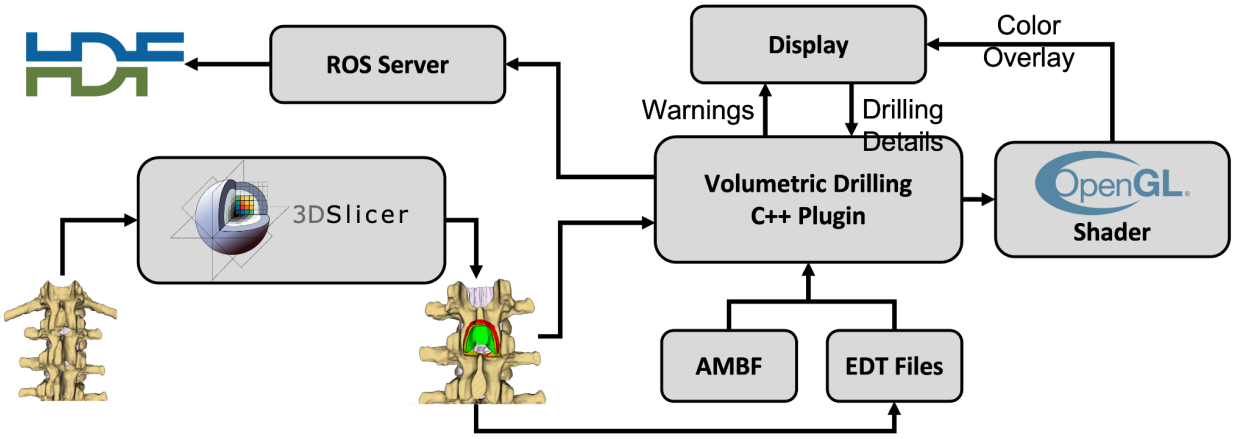
The functionality of the drilling simulator is constrained by the GPU limits of the computer used. This document further assumes all hardware dependencies are met; this will be discussed in the Dependencies section.

System Overview and Design

The VR simulator is built on top of the AMBF platform. The displayed 3D anatomies are derived from CT scans that were segmented in 3D slicer with the intended surgical cuts. The color-guidance was implemented using an OpenGL shading script that assigned colors based on the loaded SDF volumes.

Interface Diagram

The below diagram specifies how the different interfaces in the VR system work together.



Five lumbar spines were segmented in 3D Slicer with the desired drilling cuts. These segmentations were uploaded as image volumes in the {./resources} folder using the Python script seg_nrrd_to_pngs.py. SDF volumes were generated from the image volumes using an offline C++ pipeline. At initialization, a graphical user interface (GUI) is launched from *./scripts/study_gui/study_gui.py* that lists all available anatomical scans for users to practice drilling (Figure 6). Once a spine and region is selected (e.g., spine P1, region L1), the AMBF simulator is run from *./plugin/volumetric_drilling/\\volumetric_drilling.cpp* to render this case. The composite SDF is based on distances from identified anatomies and structures that should not be drilled. The color overlay uses the composite SDF to caution users against drilling too far in any coordinate direction. Red regions highlight bone that is less than 2mm away from a sensitive anatomy, yellow regions color bone that is between 2mm and 4mm away, and green regions indicate bone that is greater than 4mm away. Text-based warnings are thrown based on whether the drill tip, in the coordinate frame of the simulator, is within either the yellow or red

regions. These warnings are positioned to the side, beyond the central field of view, to limit occlusion.

If the recording button is pressed, the VR platform begins recording simulation data (e.g., coordinate and color of voxels removed, drill kinematics, drilling time) and publishing it to a ROS server. A parallel script *./scripts/study_gui/data_record.py* subscribes to the ROS server, queries the data, and locally saves it to HDF5 files (in chunks of 500). This asynchronous data extraction is a burden on the PC and contributes to laggy user sessions after extended drills; our technical developments tackle this issue with a proposed synchronized C++ plugin.

The simulator requires specific hardware components: a GPU-accelerated Linux machine to run the AMBF simulator software, HTC Vive Pro VR headset, Phantom Omni haptic device, headphones, and keyboard. Users are seated in front of the monitor and wear the headset to navigate the virtual world. The haptic device is used to manipulate the drill. A drilling sound will be played and the cutting force is used to set the signal pitch.

Dependencies

Hardware dependencies:

Dependency	Need	Contingency	Status	Planned	Resolved
Linux Machine	To run AMBF and build VR simulator locally to do technical development	Borrow Gaming Laptop form Hisashi (not powerful enough for data recording)	Resolved	3/6	3/13
HTC VIVE Headset	To test color shading display for participants	N/A	Resolved	3/6	3/13
Phantom OMNI Haptic Device	Test warning and haptic feedback for participants	N/A	Resolved	3/6	3/13
Headphones	Provide audio feedback to participants	Purchase headphones outside	Resolved	3/15	4/5

Software dependencies:

Dependency	Need	Contingency	Status	Planned	Resolved

ROS	To subscribe to ROS server and perform data-analysis	Rely on synchronous data extraction	Resolved	3/15	3/14
AMBF	Backbone for the entire VR drilling simulator	N/A	Resolved	3/15	3/14
HDF5 C++ Library	Synchronized data extraction from within volumetric	N/A	Resolved	5/4	5/4
Phantom OMNI Drivers	Connect the OMNI device to be detected in the simulator	N/A	Resolved	3/25	3/24
OpenGL Package	Display the color shadings on segmentations	N/A	Resolved	3/22	3/23

Functional Requirements

Data

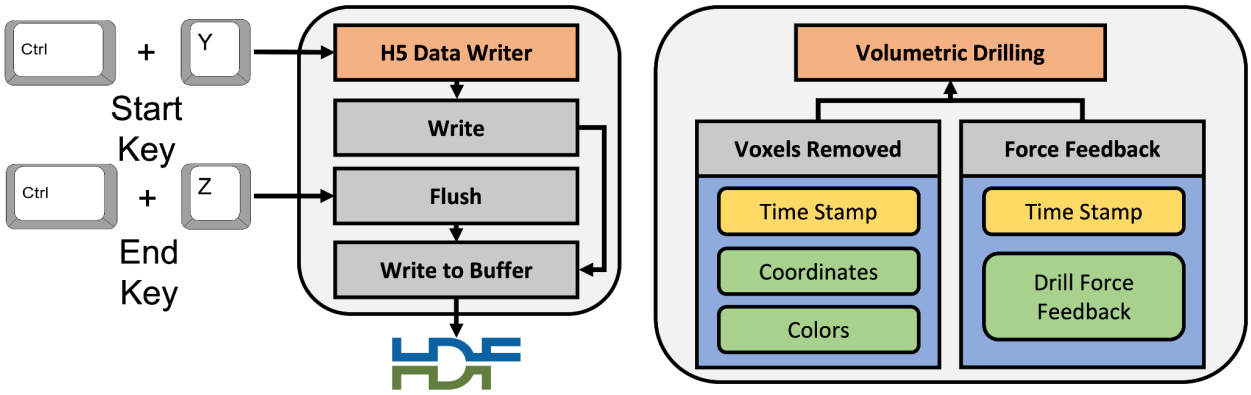
Requirement: Rendered 3D anatomies need to precisely represent true lumbar spines seen by surgeons intraoperatively.

Solution: CT scans of the lumbar spines of five patients were acquired from the Johns Hopkins Medical Institute. These were extracted into PNG files (each plane along the z-axis was converted to a 2D image) using 3D slicer and Python.

Requirement: In order to evaluate the efficacy of color-guidance, user sessions in the simulator need to be analyzed for drilling accuracy, drilling time, and number of breaches into sensitive anatomies.

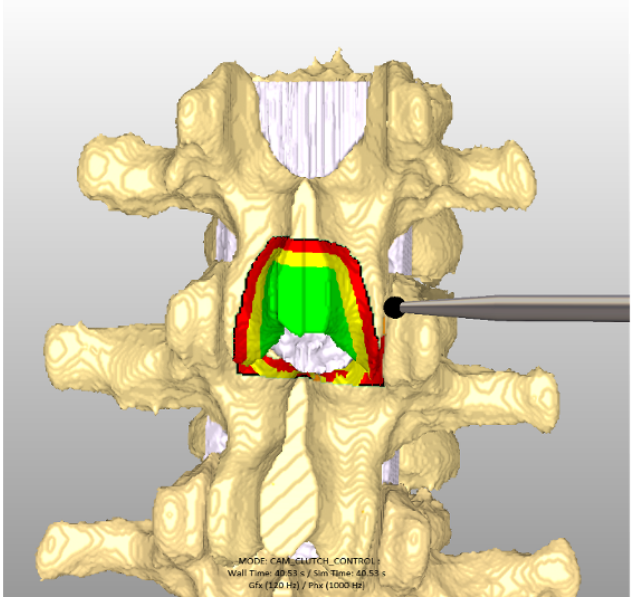
Current Solution: The system currently records data using a ROS server. When recording, the VolumetricDrilling C++ plugin publishes drill kinematics and voxel removed coordinates to the server. The plugin is built on top of AMBF, which tracks object position/movements. This data is extracted using a parallel data analysis script written in Python and saved locally as HDF5 files.

Improved Solution: Subsequent iterations of this simulator will move the HDF5 saving to a native C++ class inside the plugin. A diagram of this interface is provided below.



Visual Rendering

Requirement: Color overlay needs to guide users both radially and depthwise, as shown in the figure below.



Solution: To form the composite EDT on a particular lumbar vertebrae, three individual EDTs were joined together: the posterior spinal cord, the bone surrounding the lamina to be removed, and bone from the immediately below vertebral region. Taking the minimum of these three EDTs allowed us to form the requested visual guidance and throw warnings when users moved the drill too far in any coordinate direction.

Requirement: Text-warnings need to be less distracting for users while drilling.
Solution: Text boxes are positioned outside the field of view. Color overlay is provided on the anatomy to make a less cognitively demanding display.

Requirement: Scene lighting needs to reflect operating room lighting in order to create an immersive user experience.
Solution: Light sources were positioned in the OpenGL shader and ray-casting algorithms were used to appropriately light and shadow different structures. See the code for documentation.

Requirement: Users need to know whether or not they are drilling bone and how far they are from sensitive anatomies (e.g., the spinal cord).
Solution: Utilize signed-distance fields to generate SDF fields that compute relative distances from the drill tip to different anatomies.

$$g_{ij} = min_x\{(i - x)^2; f_{xj} = 0; 1 \leq x \leq L\}$$
$$h_{ij} = min_y\{g_{iy} + (j - y)^2; 1 \leq y \leq M\}$$

Equation 1: Two-step transform in the 2D Euclidean Transform used to compute the SDF grid. In 3D, this would involve a z-axis minimization.

$$S^{composite}(i, j, k;\ L_i) = min\{S^{VF}(i, j, k),\ S^{L_i}(i, j, k),\ S^{L_{i+1}}(i, j, k)\}$$

Equation 2: Composite SDF computed based on signed distances from the VF, bone surrounding the current lamina (L_i region), and bone in the lumbar vertebra beneath the lamina (L_{i+1} region).

Haptic Feedback

Requirement: Users need to feel that they are drilling out bone versus drilling air.
Solution: Utilize the Phantom Omni device to have users control a drill-like object while also being able to sense haptic feedback. Additionally, audio pitch changes based on whether the drill tip is milling bone or air.