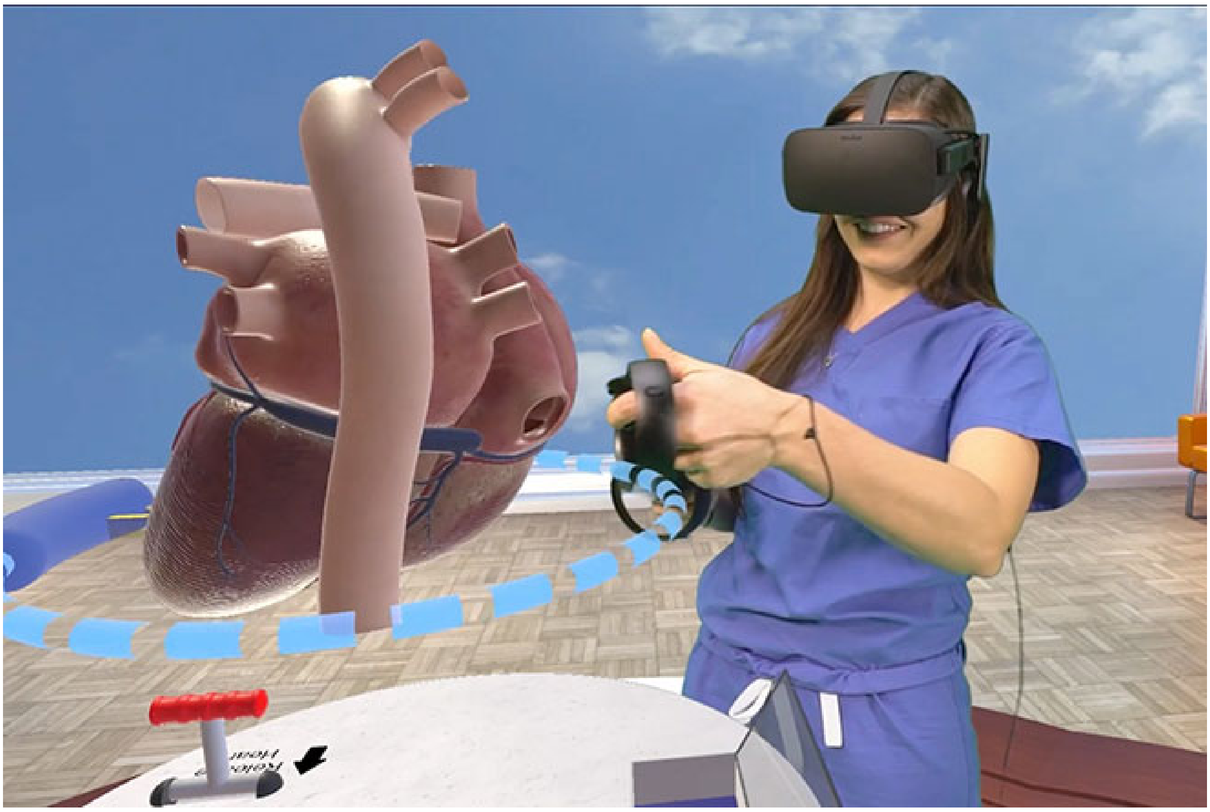
ICME Xplore Project Final Report

Stanford Virtual Heart - Blood Flow System

Mentors: David Sarno, David Axelrod — Jun Li
Faculty Sponsor: Professor Alison Marsden — Autumn 2018

Project Background



The flow of blood through the heart is fundamental to cardiology and cardiovascular physiology. When patients have cardiovascular diseases – either congenital or adult – the result is that the blood doesn't circulate correctly, and oxygenated blood does not adequately reach the body organs or the heart itself. For this reason, an accurate and compelling depiction of blood flow in the SVH is crucial.

The SVH previously used a bespoke system to generate animations of blood flow, which includes a set of handmade spline curves along which float 3D models of blood cells. The goal is to overhaul the blood flow system to be more accurate, visually appealing, and flexible. Ideally a new bloodflow system would be authorable, allowing designers and non-coders to easily create new bloodflows within Unity.

Throughout this project I made two main improvements: (1) Effectively combined the realistic blood flow simulation data from SimVascular with the virtual heart VR/CGI model. (2) Simplified the process pipeline to be used for future development.

Previous Work

(1) Unity model: A virtual heart model in Unity was provided by Lighthaus Inc. (`https://youtu.be/xW1EMBVmAW4`)
(2) Blood Flow Simulation: Previous work by Kyle Qian was provided, which includes the following:

a. Tutorials for a basic SimVascular simulation (`https://youtu.be/szqzWegd5EA`) and Paraview visualization (`https://youtu.be/4iOXas2ukgc`).

b. Blood flow visualization in Unity as flow lines (`https://youtu.be/4AbRMlkPhCw`). This visualization requires almost 30GB of .fbx files data exported from SimVascular simulation thus results in an inefficient workflow.

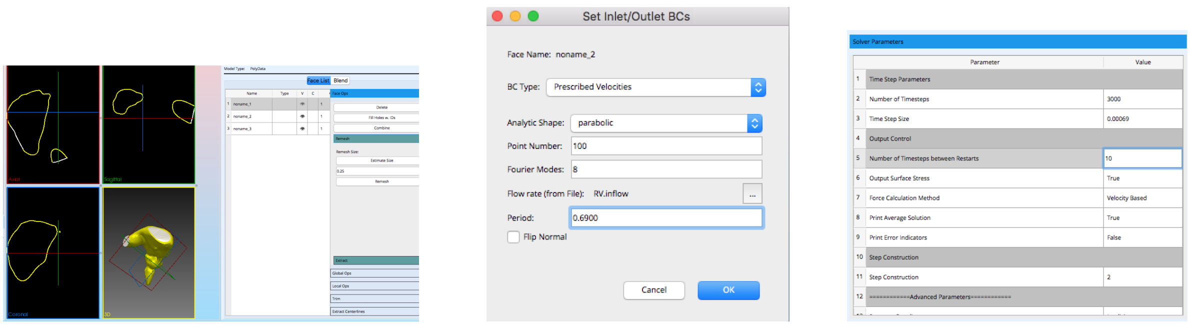
c. Particle tracing prototype in Unity using VMTK (`https://youtu.be/O26B9WrOcuI`). Kyle mentioned that VMTK was hard to work with, and it didn't work on my device, so I later switched to another software, flowVC for better functionality and workflow.

A complete documentation of Kyle's work can be found **here**.

Improved Pipeline

Step 1 - SimVascular

Input the .stl model, inflow text file, and correct parameters to setup the simulation. Output .vtu files as the velocity field for simulation, with the amount being the number of time steps.



(a) .stl model import (b) simulation inflow setup (c) simulation time step setup

Figure 1: SimVascular simulation setup

Step 2 - Paraview

Input the .vtu files from last step and directly convert to the corresponding .vtk files. At the meantime Paraview can be used to visualize and check correctness of the simulation from step 1.

Step 3.1 - flowVC - vtk2bin

Input the .vtk files from last step and one of the following: (1) converted .bin files using SimVascular 1.0 (.vtk to .vis) and old version flowVC (.vis to .bin), or (2) a '.xadj' file and a '.connectivity' file generated from my script generateAdjacency.py. Note that the flowVC output for this method doesn't include the velocity normal bin files thus the

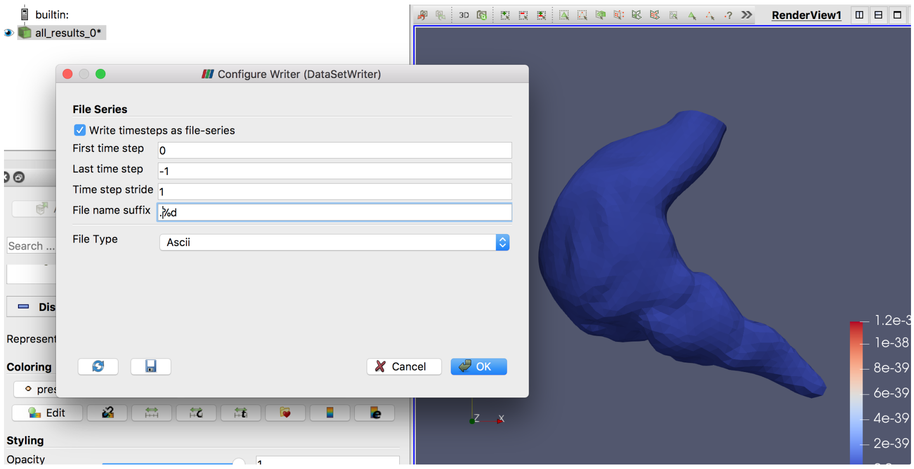
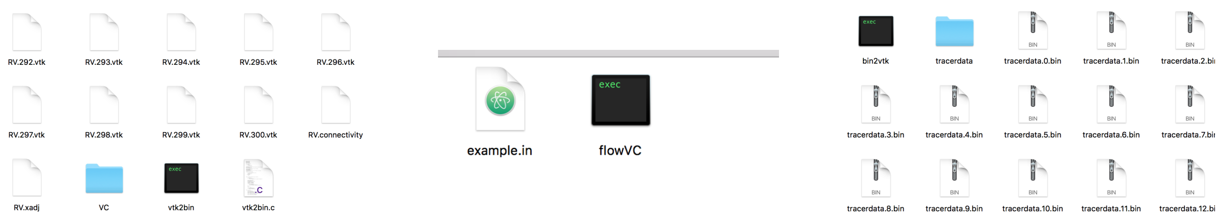


Figure 2: Paraview .vtk file export

result will have problem with particles stuck on the wall.
This step will output the bin files for velocities, velocity normals, and coordinates, each has the amount of the number of total time steps from the simulation.

Step 3.2 - flowVC - flowVC
Input all the .bin files from last step and a setup file for flowVC (example.in), and output the processed .bin files for tracerdata with particle positions.

Step 3.3 - flowVC - bin2vtk
Input the .bin files for tracerdata from last step, and output the corresponding .vtk files for tracerdata.

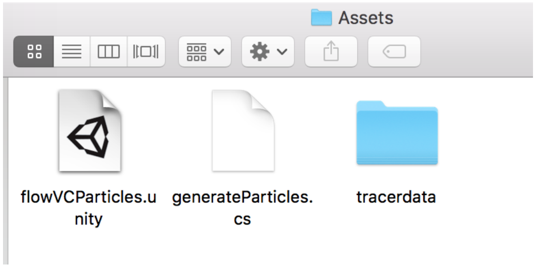


(a) 3.1 vtk2bin conversion (b) 3.2 flowVC data processing (c) 3.3 bin2vtk conversion

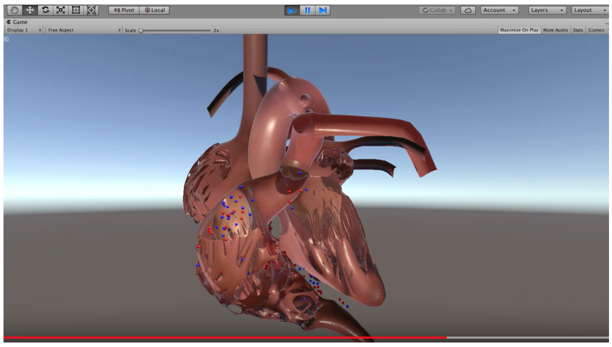
Figure 3: 3-step flowVC conversion

Step 4 - Unity
Add all the .vtk files for tracerdata and my script generateParticles.cs to any GameObject. Adjust the position and orientation to integrate with the original heart model.

Final Result
Blood flow as particles integrated with the test heart model: `https://youtu.be/PB-PiYacVd0`.



(a) Unity data import



(b) Unity particle simulation

Figure 4: Unity visualization

Future Steps

1. Record a video tutorial for these steps.
2. Polish on visualization, especially on cycle/position match, blood cell model, cell number and looping simulation.
3. Find a way to generate the missing velocity normal files from flowVC vtk2bin conversion.

Takeaway

This project has been really meaningful and rewarding to me. The idea of combining VR technology and medical education constantly amazes me and throughout the process I learned to work on both the academic side and the more client-oriented side of such a project. The scope of the project is much larger than a class project and includes much more hardware and software setup. It was really helpful to meet with my mentor weekly to learn about the context and previous work, as well as sharing the difficulties I was facing. As the project involves ideas and work from many different perspectives, the challenges I had were also different in that this real world project has more problems related to software accessibility, people's availability, and ambiguity in the approach I should take.

I found it very helpful to work with people with other specialization of knowledge. Especially during the step switching VMTK to flowVC, I was stuck for a while and almost tried to write my own version with the help of Vijay. At that point it was difficult for me to move forward without any knowledge in related area. After getting in touch with more people in the lab and the author for the software I got information and instruction on those steps that I would probably never figure out myself (the complete version for flowVC, the problem caused by the lack of .xdj and .connectivity files, etc).

I really appreciate the opportunity to work on this fascinating project with many supportive and helpful people. I would love to keep working on the next step or helping people who work on it.

Appendix

1. All files used for the simulation and final result:
   `https://drive.google.com/drive/u/1/folders/1QFc15tKe_k30h33TF8EmRrw6q_sdrL6L`.

   SVH Blood Flow Jun Li.zip: includes all simulation files with 2 ways to convert results using old/new flowVC, and a Unity project for particle demo.
   SVH_2018_10_15_Jun Li.zip: includes blood particles integrated with the original SVH project heart model from Assets/Tests/BloodflowTest. The only added files are in Assets/Scripts/BloodFlow/genearteParticles.cs+tracerdata+tracerdatanew. Note that tracerdata is currently used for a simulation without particles stuck on the wall.

2. A list of software needed for the project:
   SimVascular: `http://simvascular.github.io`
   Paraview: `https://www.paraview.org/download/`
   flowVC: `http://shaddenlab.berkeley.edu/software.html`
   (the vtk2bin file is not on the website - there is a copy in my project folder.)
   Unity3D: `https://unity3d.com/unity`

3. A list of people involved in this project:
   Lighthaus Inc: David Sarno, Prof. David Axelrod
   Marsden Lab: Prof. Alison Marsden, Justin Tran, Vijay Vedula, Weiguang Yang
   Shadden Lab: Prof. Shawn Shadden
   Previous work: Kyle Qian