3D Reconstruction using Time of Flight Sensors

Mentor: Mani Mina
Advisor: Professor Daniels
Client: VirtuSense Technologies

Team Dec15-09:
Monica Kozbial
Kyle Williams
Sarah Files
Yee Zhian, Liew

Overview

- VirtuSense Technologies
- Project Summary & Market study
- Project Phases
- Current Progress
- Project Detail
- Challenges
- For Next Semester

VirtuSense Technologies

- Markets innovative solutions to healthcare providers including:
  - VirtuOR: Monitors the operating room to determine how time can be better used
  - VirtuBalance: Provides data to reduce risk of fall for patients
  - DyST: Analyzes athlete performance and provides feedback

Project Summary

- Requested by VirtuSense Technologies
- Target User: Cosmetic Surgeons
- Simulates the effect of cosmetic procedures on a patient's face
- Utilizes the Kinect version 2.0 "time of flight" sensors

Project Phase Overview

Three Phases:

- Phase 1: Capture the Model
- Phase 2: Edit in Blender
- Phase 3 (Stretch Goal): Full Body Scan

Phase 1: High Quality Model

Use Kinect 2.0 for Windows 8 to create a 3D model

-Capture the subject's face and convert to a 3D model

-Apply texture overlay

-Export to Blender

Deliverables

1. Converting 3D models to 3D meshes
2. Smoothing algorithms for 3D meshes
3. Texture overlay on the 3D models

Phase 2: Edit in Blender

Once the 3D Model is in Blender, create a user friendly UI for manipulation

-Create an add-on that limits tools to the essentials

-Develop 3D morphing algorithms to manipulate any selected meshes on model

Deliverables

4. UI for manipulating 3D models
5. UI for selecting individual regions from a 3D model
6. Algorithms for 3D morphing both on meshes and textures

Phase 3 (Stretch Goal)

Scale Phase 1 to allow full body scans

-Instead of just the face, create 3D model from entire body

-Only if enough time after Phase 1 and 2

Deliverables

7. Geometry calculations for locating sensors for whole body capture
8. Algorithms for 3D morphing on selected regions on the whole body

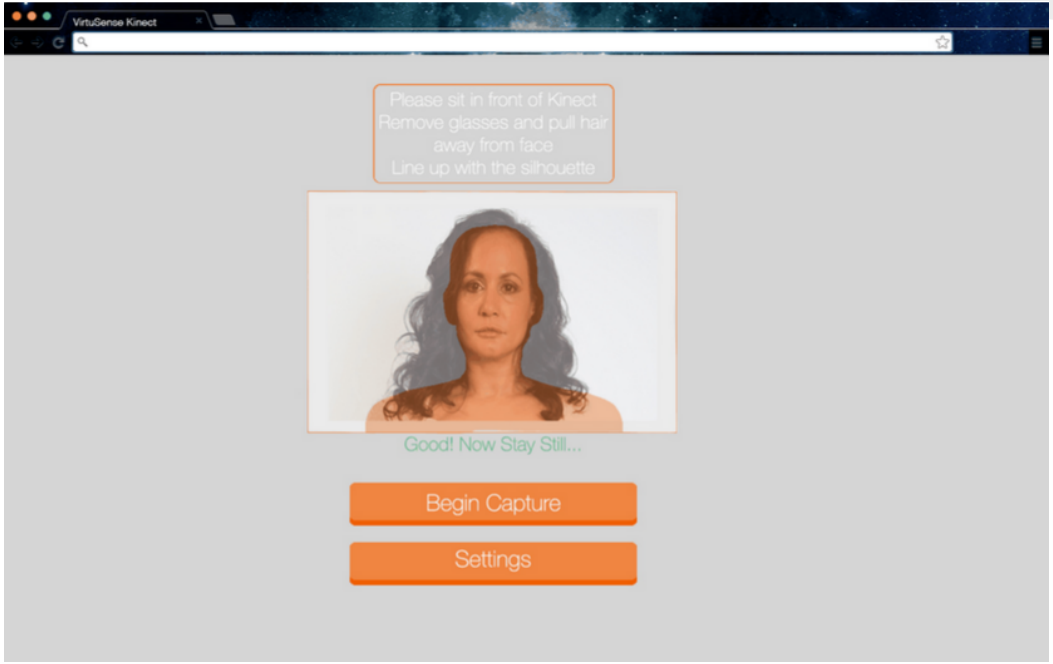
Current Progress

Currently in Phase 1:

- UI design
- Texture mapping
- FaceModelBuilder and HDFace
- Improving Model
- Kinect sensor pipeline

Kinect User Interface

- Web Application
- Responsive web page
- Works with Kinect to capture model
- Local Program, no internet required
- Export captured 3D model to Blender

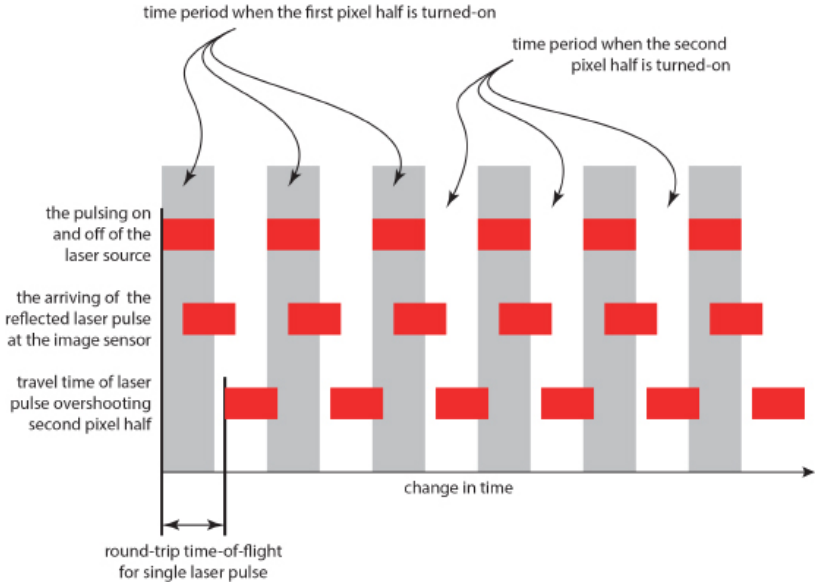


Kinect Version 2.0 Sensor

Specifications

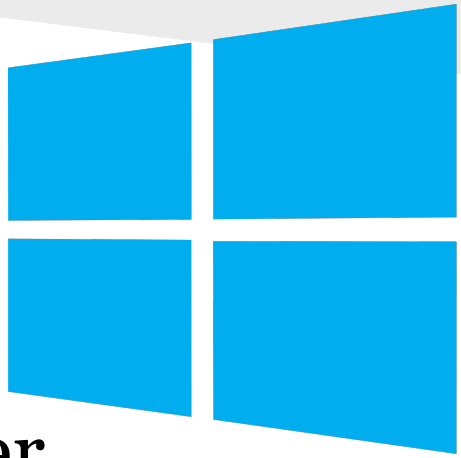
	Kinect for Windows v1	Kinect for Windows v2
Color	640 × 480 @ 30fps	1920 × 1080 @ 30fps
Depth	320 × 240 @ 30fps	512 × 424 @ 30fps
Sensor	Structured Light (PrimeSense Light Coding)	Time of Flight (ToF)
Range	0.8~4.0 m	0.5~4.5 m
Angle of View Horizontal / Vertical	57 / 43 degree	70 / 60 degree
Microphone Array	○	○

Time-of-flight Technology



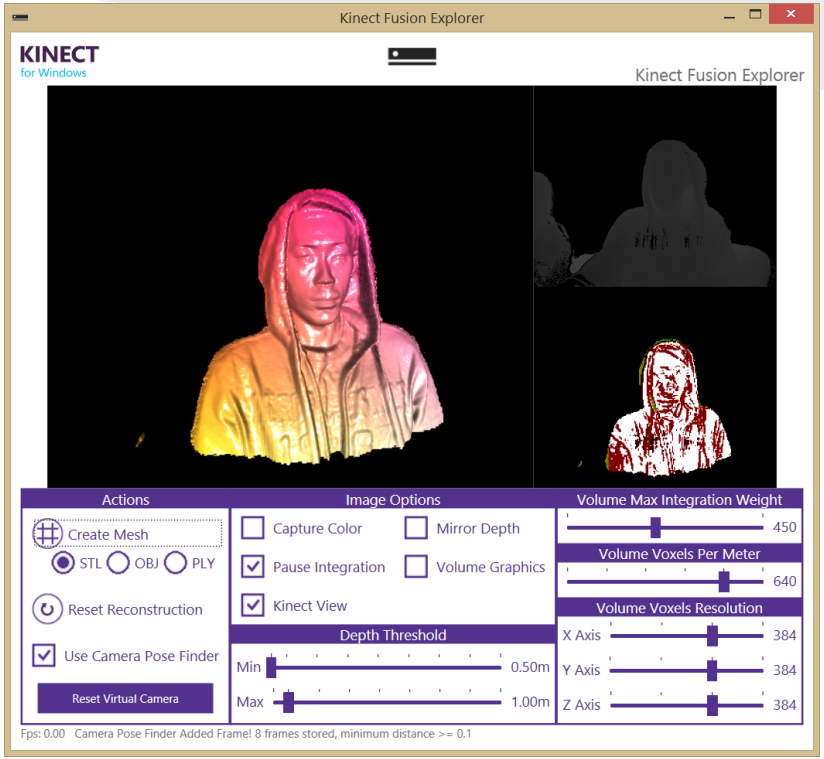
System Specifications

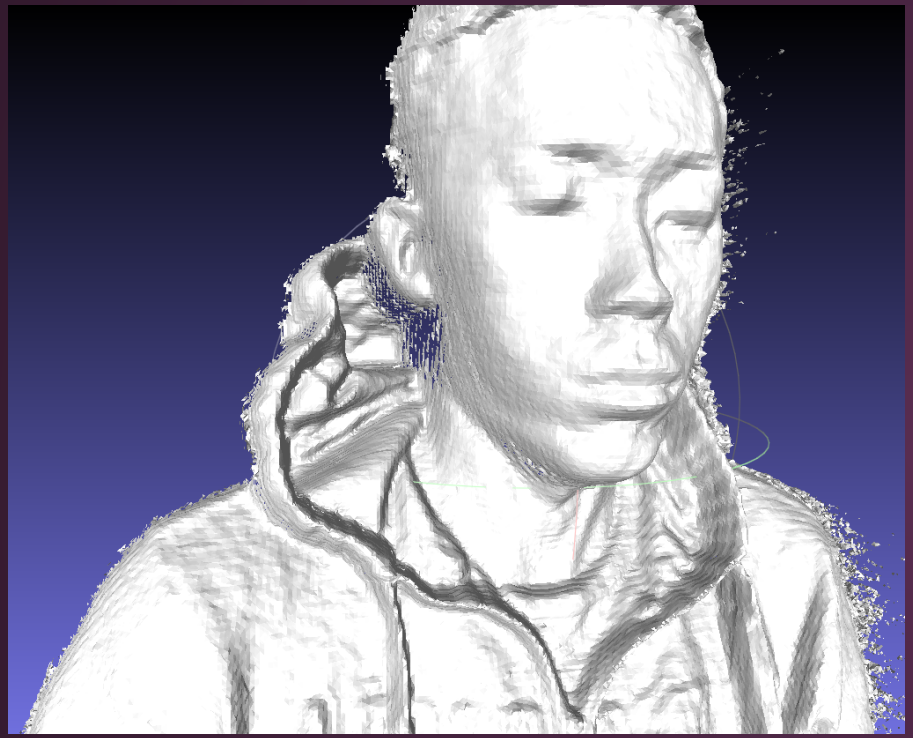
- Windows 8/8.1
- 64 bit (x64) processor
- 4 GB Memory (or more)
- i7 3.1 GHz (or higher)
- Built-in USB 3.0 host controller
- DX11 capable graphics adapter

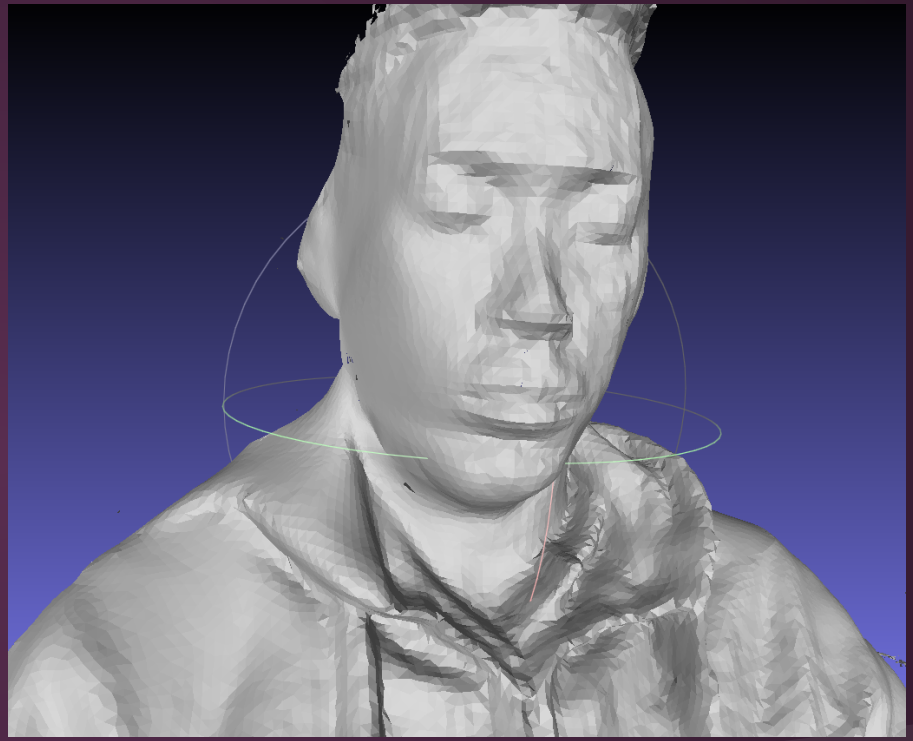


KinectFusion

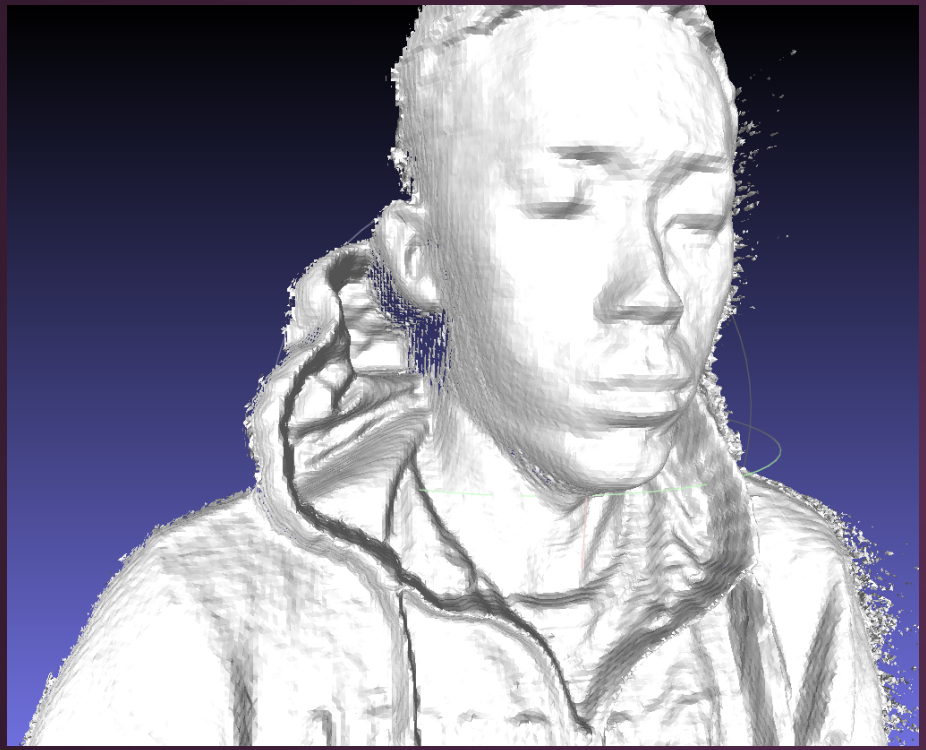
- Maximum Integration Weight
  - controls the temporal averaging of data into the reconstruction volume
- Depth Threshold
  - determines the region of the reconstion volume
- Volume Voxels per Meter
  - scales the size that a voxel represents in the real world

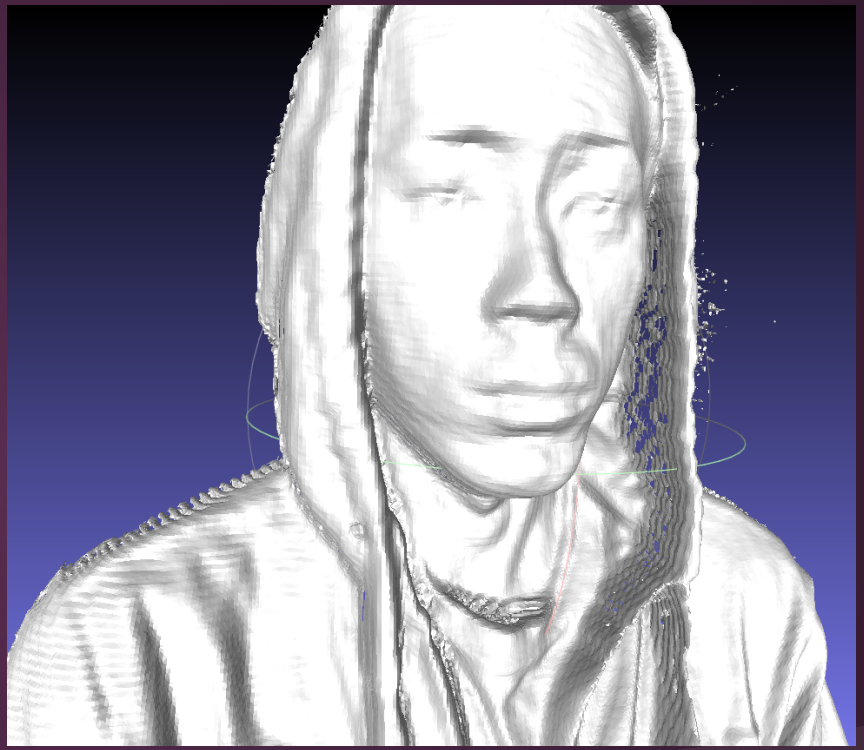




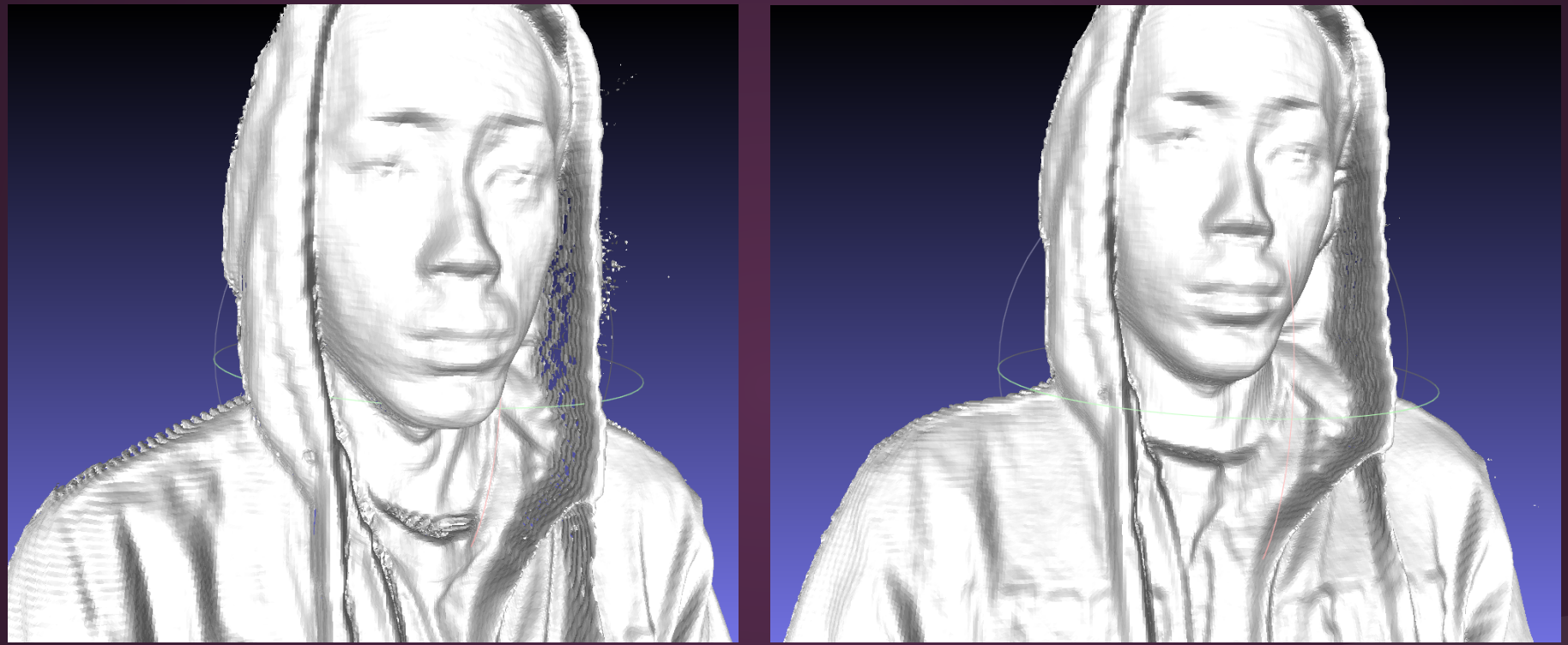


Smoothing Attempts





Early Parameter Testing (left) vs Recent Parameter Testing (right)



Intel HD Graphics Family (left) vs Nvidia GeForce GT 525M (right)

Scanned model with color mapping

HD Face

Face capturing in kinect

- captures the face in 16 frames (splits into 4 regions)
- Takes 94 vectors from these regions to apply to average face
- Create the mesh and apply it to other applications

Next Semester's Goals

Spring Semester '15

- Hardware setup
- Collect data with Kinect SDK
- Research algorithms to smooth models
- Create UI Screen sketches and web application DOM

Fall Semester '15

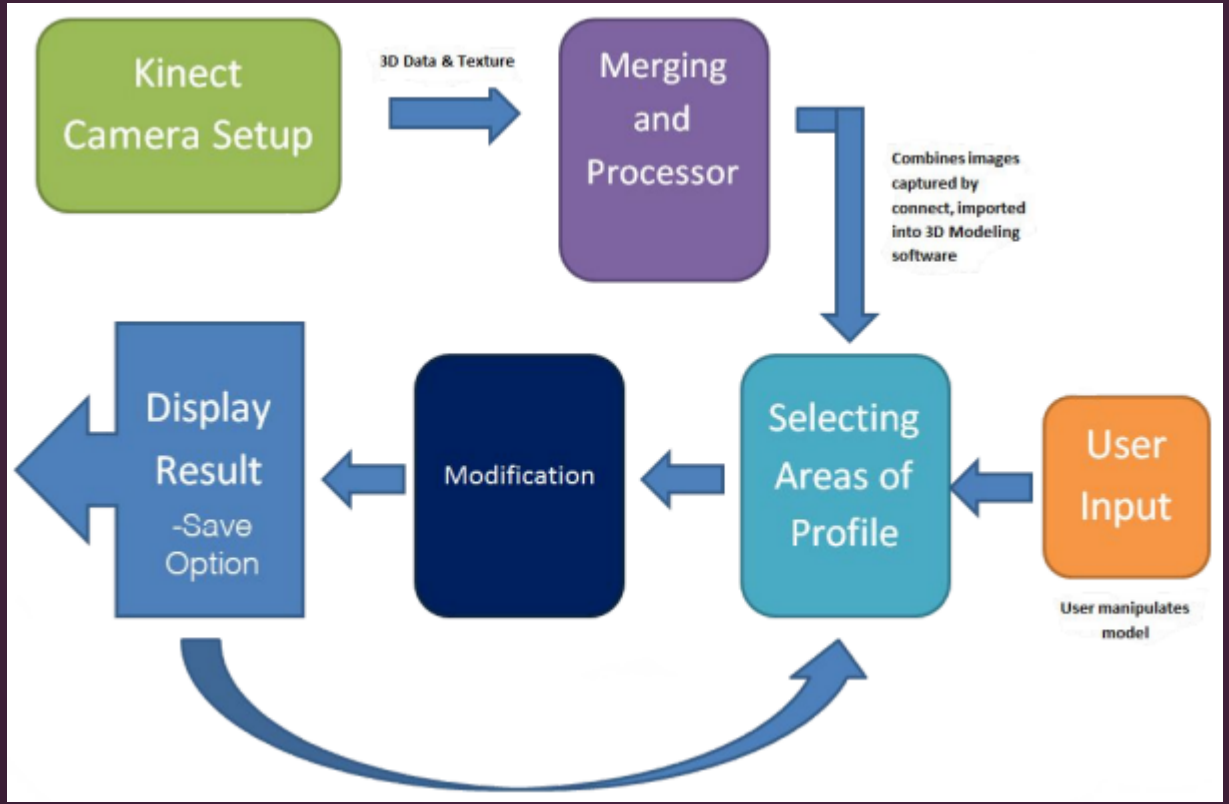
- Complete Phase 1
  - Finish refining Kinect SDK capture parameters
  - Finish implementation and Kinect UI
- Phase 2
  - UI is integrated with Blender
  - Select parts of the model within Blender
  - Apply modification algorithms

Questions?

Challenges

- Research (small area of study, limited resources)
- Blender Licensing
- Limiting Hardware/Software Requirements
  - Windows 8/ USB 3.0
  - Graphics Card
- Slow response time for resources (lab space, Kinect-ready computer, repositories, etc)
- Understanding Kinect parameters with few resources

Process Diagram Overview

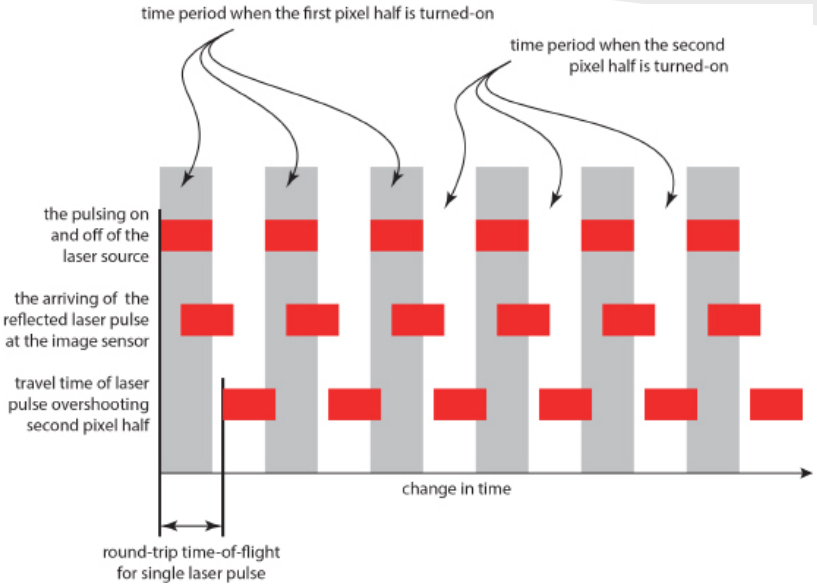


Blender Licensing

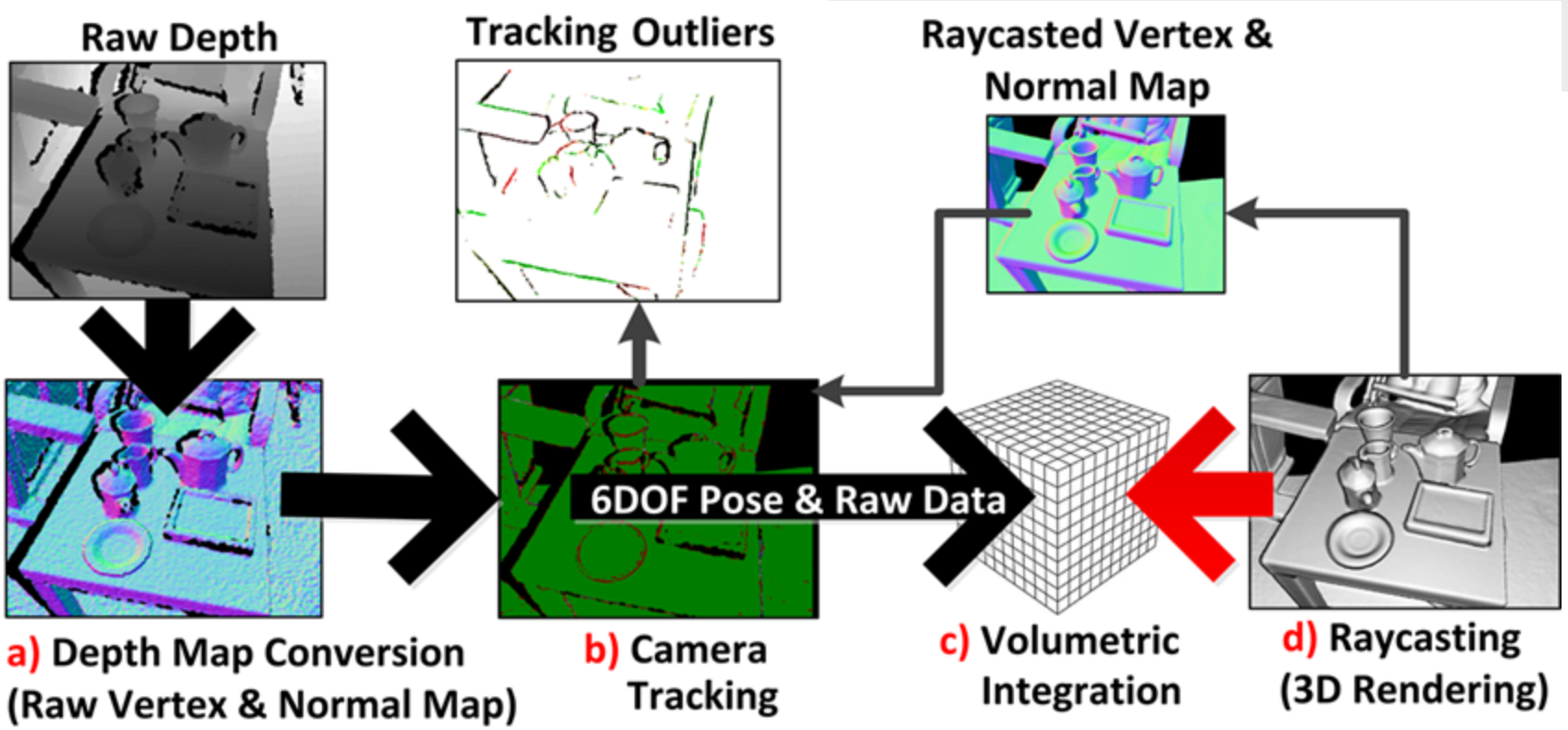
- Blender has a GNU General Public License
- A plug-in made for Blender normally must follow the GNU GPL license.
- “Only if the plug-in doesn’t work within Blender as ‘acting as a single program’ (like using fork or pipe; by only transferring data and not using each others program code) you have the full freedom to license the plug-in as you wish.” (https://www.blender.org/support/faq/)

“Time-of-Flight” Camera

A time of flight camera is a range imaging camera system that resolves distance based on the known speed of light, measuring the time of flight of a light signal between the camera and the subject for each point of the image.



Kinect Fusion Pipeline



Iterative Closest Point (ICP)

Iterative closest point finds the rotation and movement that best aligns two point clouds.