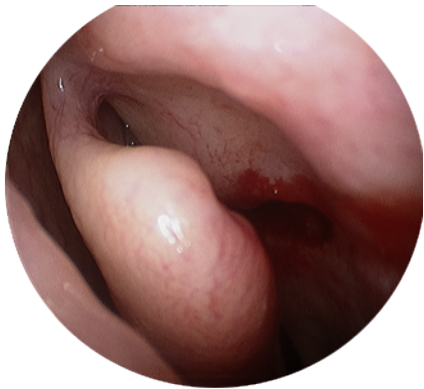# Sinus Surgery Image Processing Checkpoint Progress Report

John Lee
Kyle Xiong
Calvin Zhao
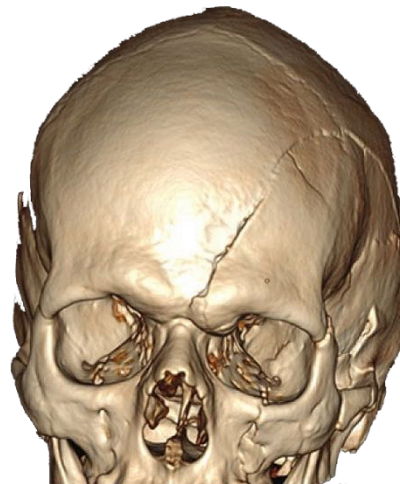
# Project Review

The goal of this project is to be able to enhance magnetic tracker registration and surgical tool position by utilizing CT data with occluding contours extracted from the endoscopic video feed.
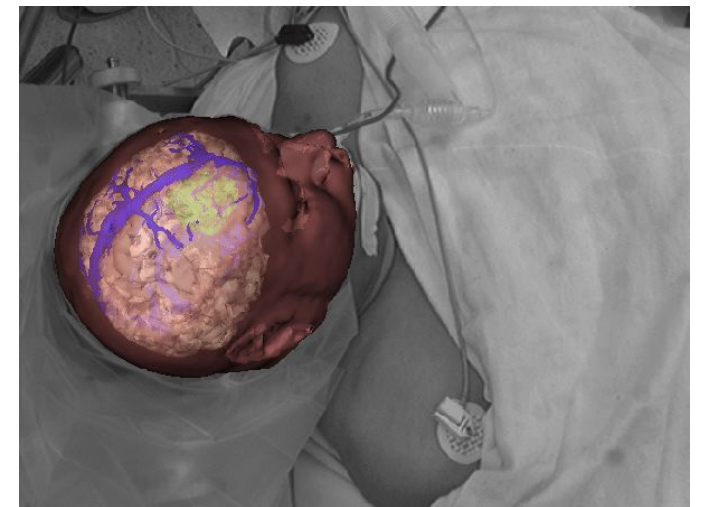
Identify Contours from surgical endoscope video feed → Integrate with CT registration developed by Seth Billings → Use augmented reality to overlay useful CT information

# Dependencies

- All dependencies are met for developing the detection algorithm.
- Our next step is to register our detected contours to CT data and achieve a high accuracy match. This depends on the registration algorithm developed by Seth Billings, which Seth has not finished developing yet.
- If this dependency cannot be met we can still develop the algorithm for contour detection, which is the main goal of our CIS project. CT registration and AR are out of the scope of the project and therefore optional.

# Deliverables

- Minimum: Develop algorithm for accurately and efficiently extracting occluding contours from surgical videos.
- Expected: A registration algorithm based on our contour detection method that can track tool-tip position in real time using only video data and CT images.
- Maximum: Augmented reality software from real-time contour detection and video-CT registration to provide surgeon with useful information.
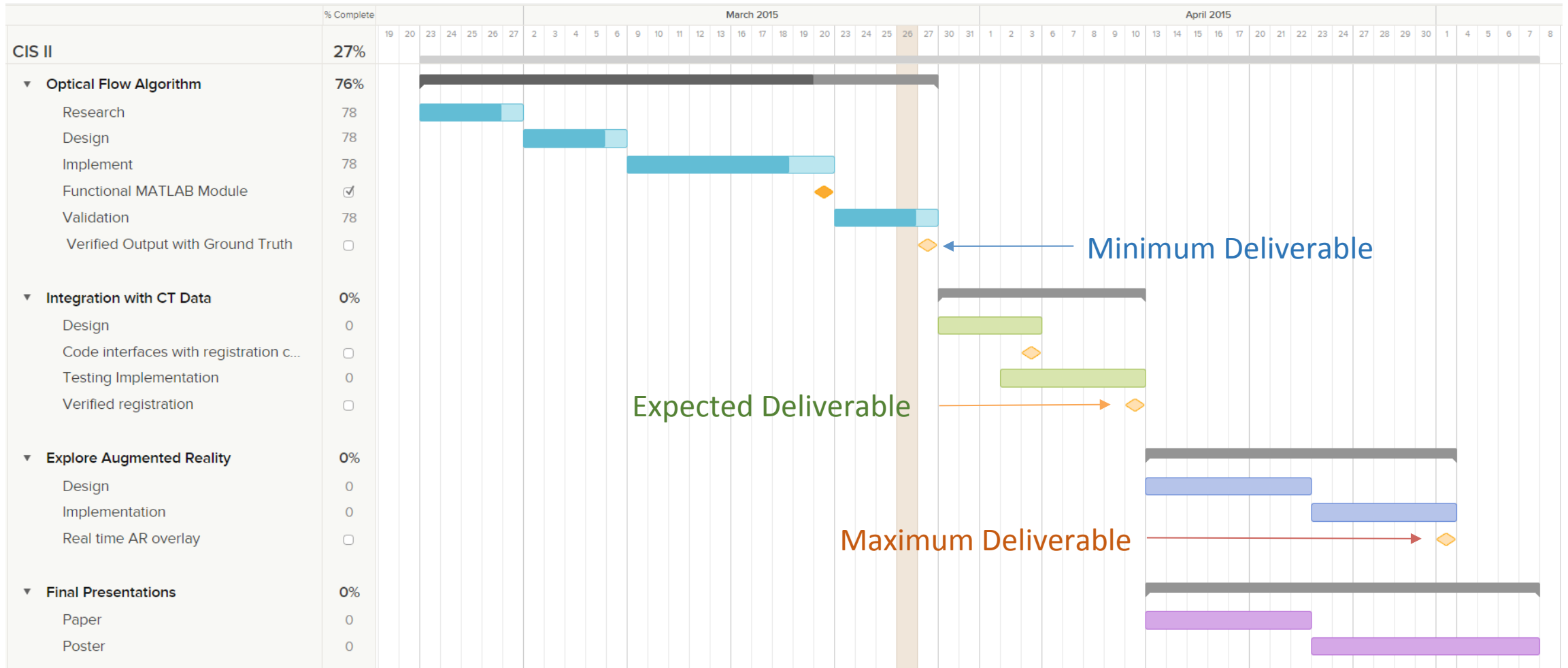
# Logistics Report

- Progress is tracked using TeamGantt
- Source control is through GitHub
- Weekly meetings are in effect
    - Mentor: Thursdays@ 10:30
    - Team members: Fridays @ 3:00

- We are confident in our minimum
- Expected deliverable is tentative
- Maximum deliverable may not be feasible

# Previous Timeline
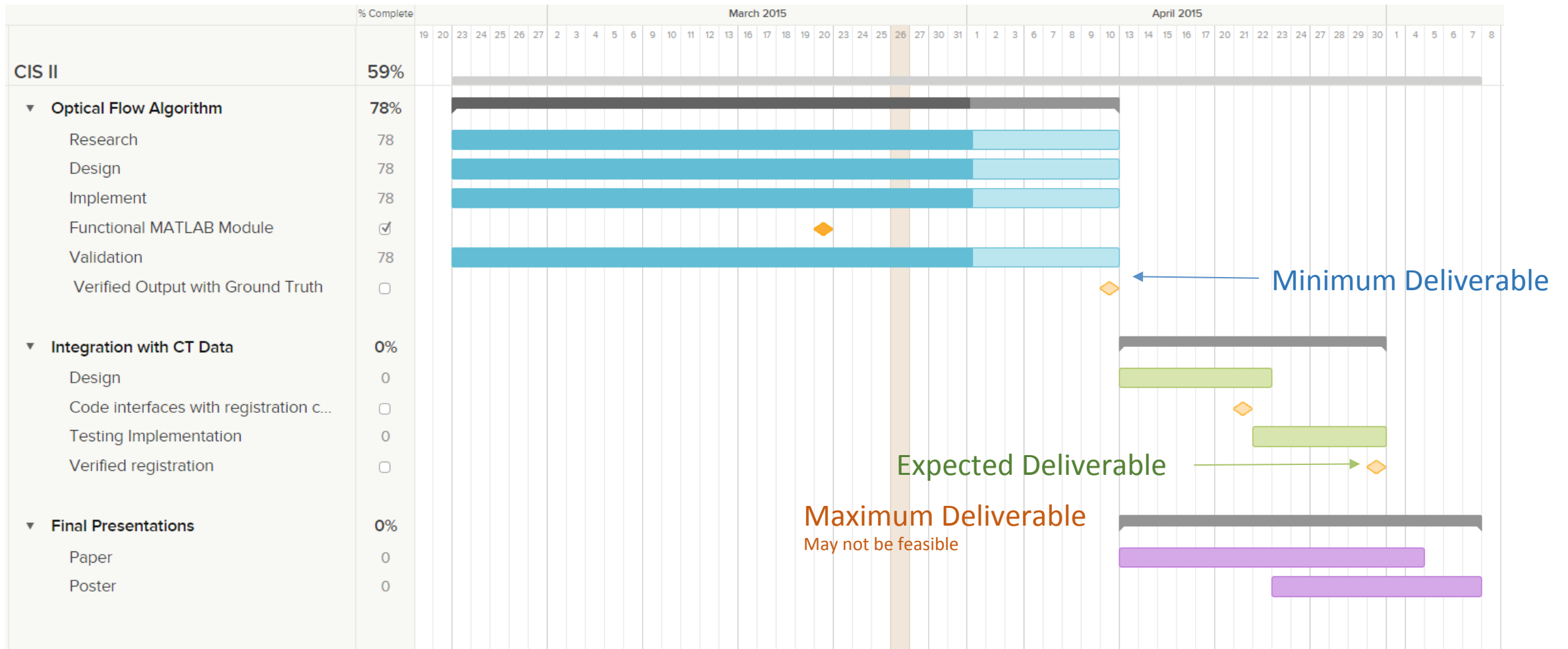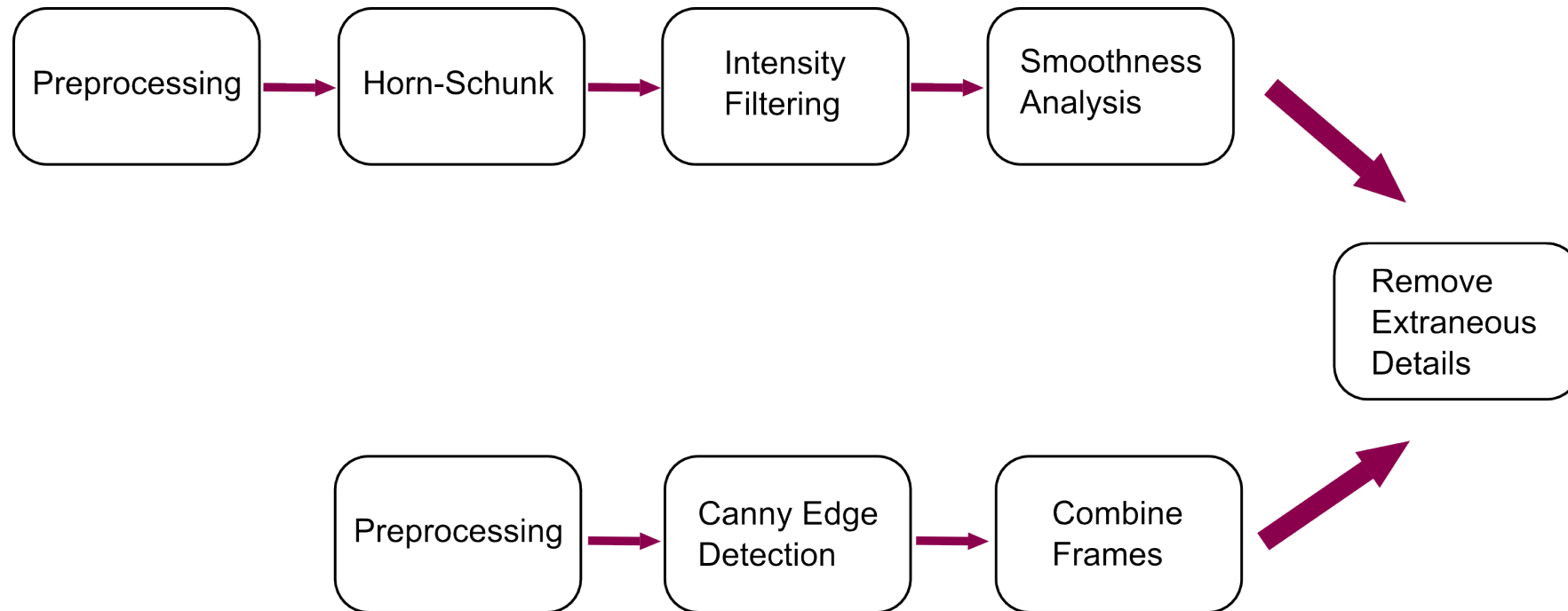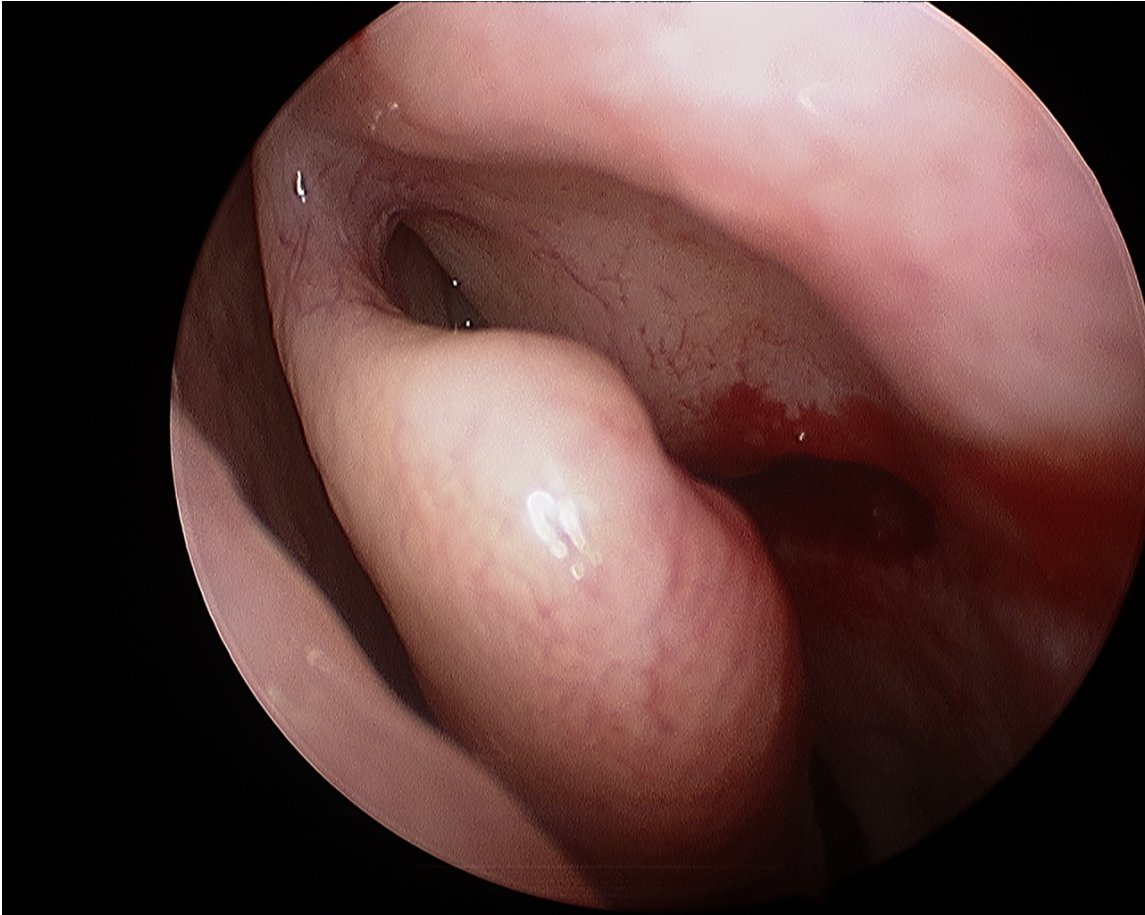
# Revised Timeline
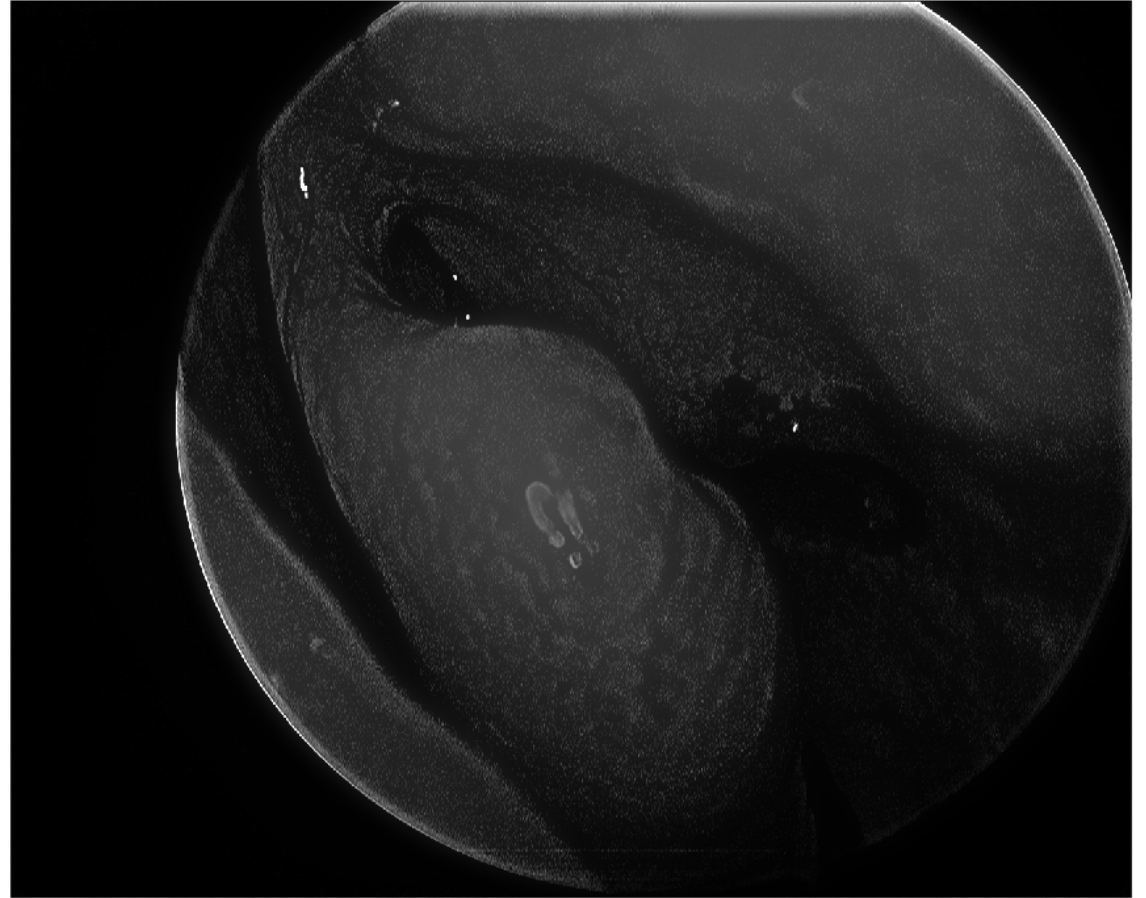
# Progress to Date

- General Workflow

# Original Image

# Edge Detection

- Preprocessing Edge Detection
  - Gaussian filter to separate high and low frequencies
  - Increases image contrast to guarantee Canny detects all edges.
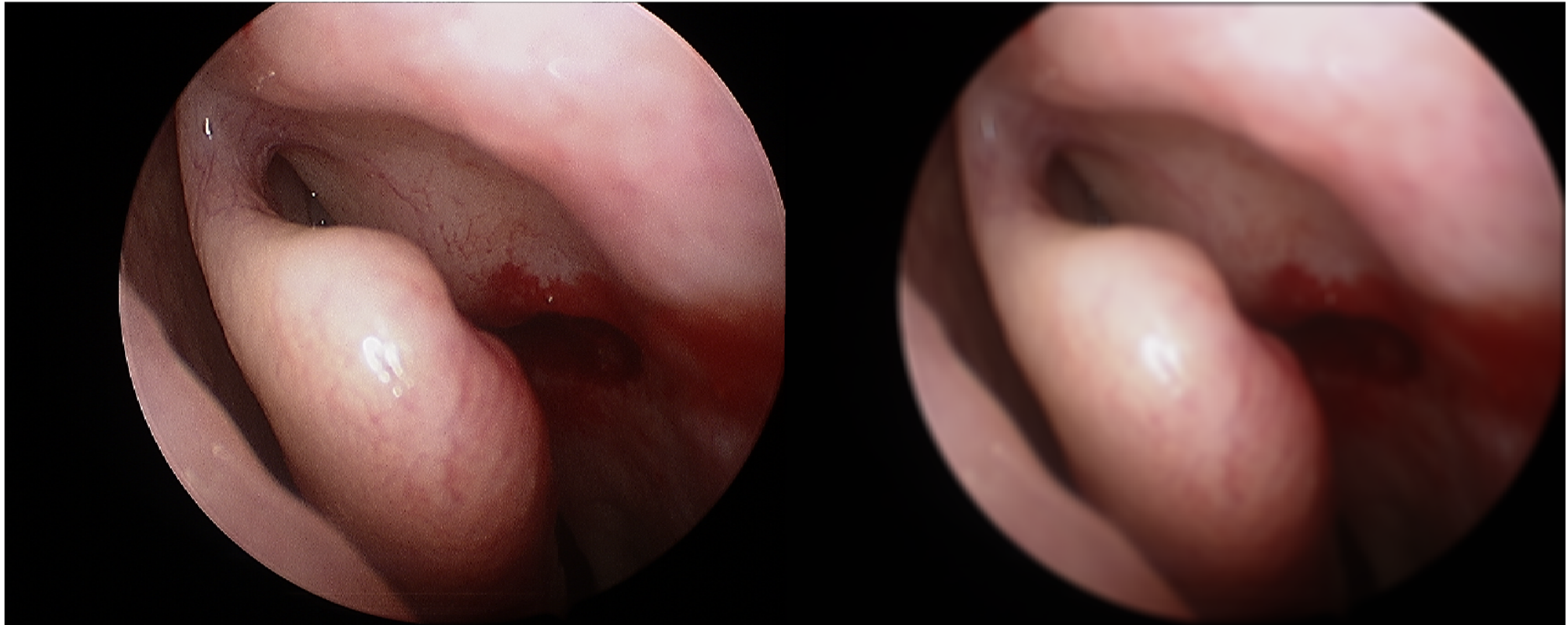
# Edge Detection

- Canny Edge Detection
  - Too much extraneous texture being detected
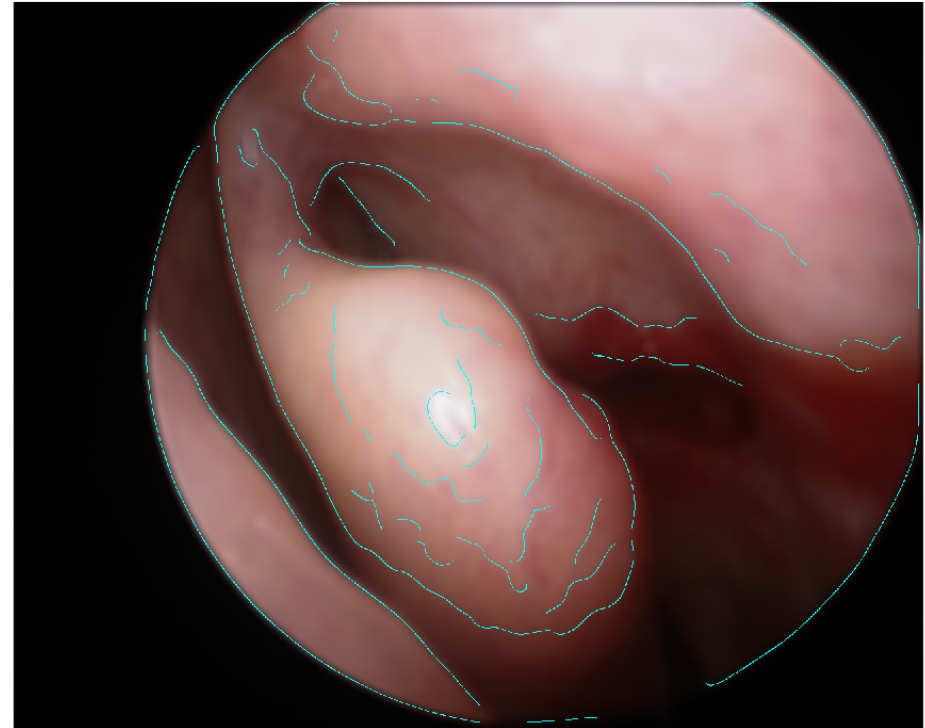  - Detects all the edges we need

# Texture Filtering

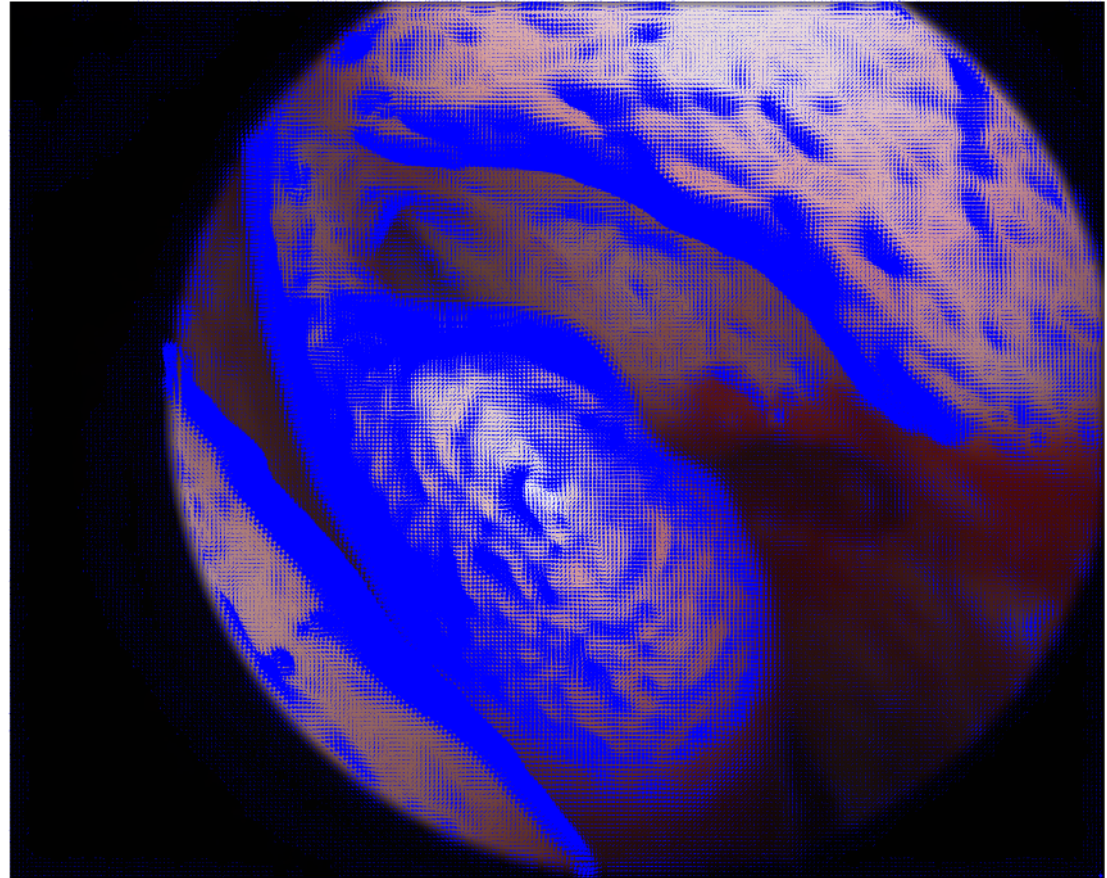- Preprocessing Optical Flow (Gaussian Blurring)

# Texture Filtering

- Simply running Canny edge detection did not return clean edges.
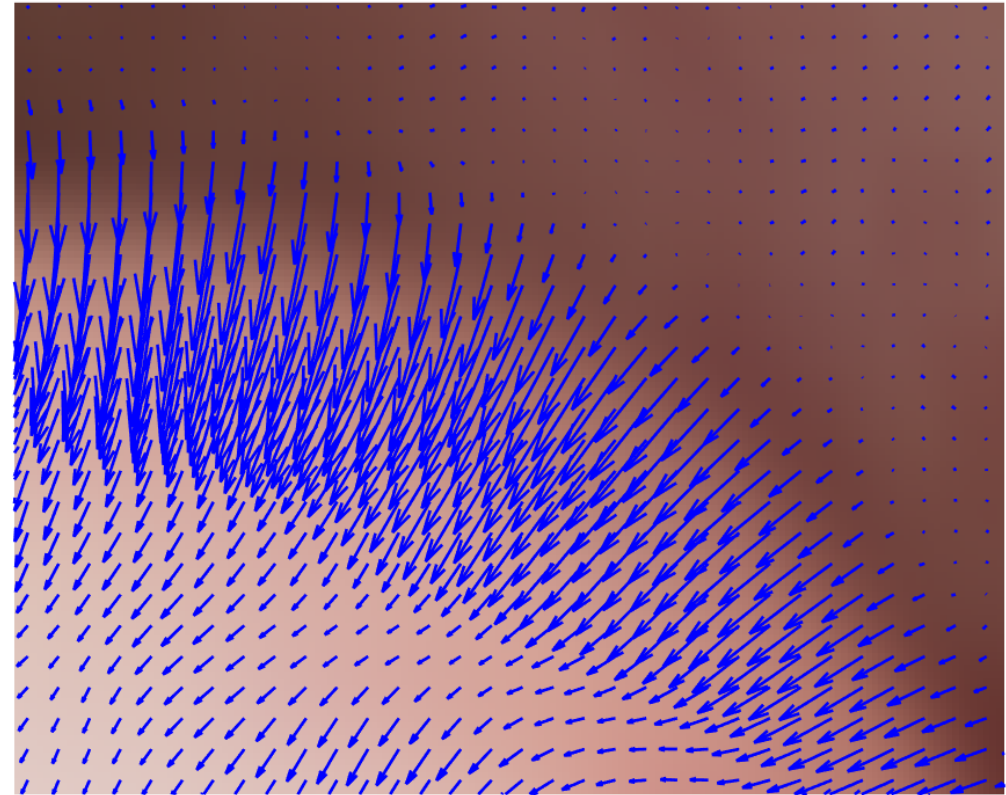- Need a way to filter out texture.

# Texture Filtering

- Motion vectors calculated using the Horn-Schunk optical algorithm.
- Vector magnitudes noticeably larger at desired edges.

# Texture Filtering

- Closer look at optical flow motion vectors.
- Uniform vectors on the edge, but uniformity fades on non-edge pixels.
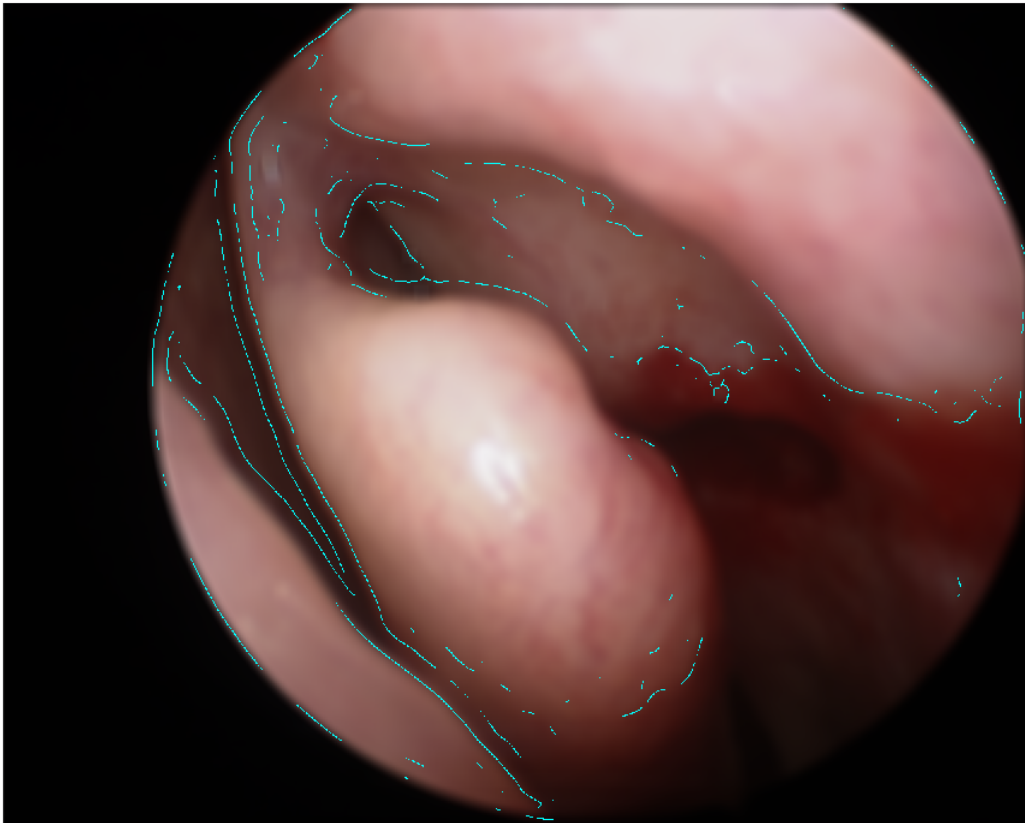
# Texture Filtering

- Calculated smoothness of motion vectors around each pixel within a certain radius.
- Smoothness inversely depends on variance of motion vector components.
- Pixels with high intensity were set to have a smoothness of 0.
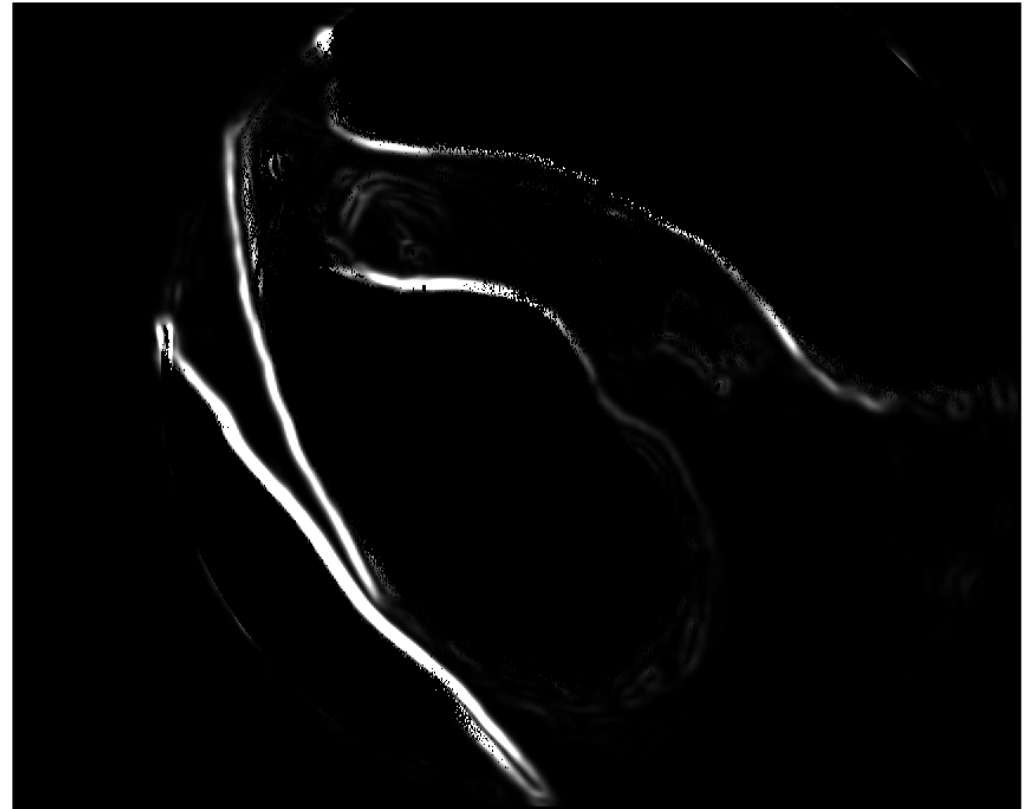
# Texture Filtering

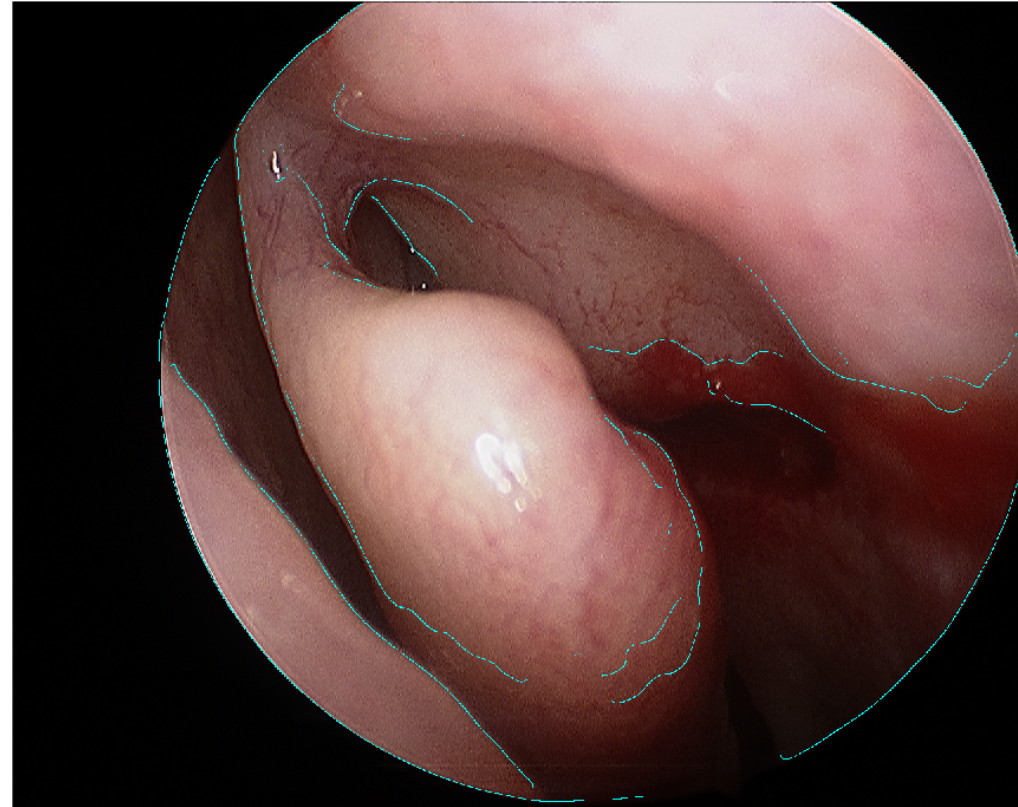- Results of multiple parameter changes
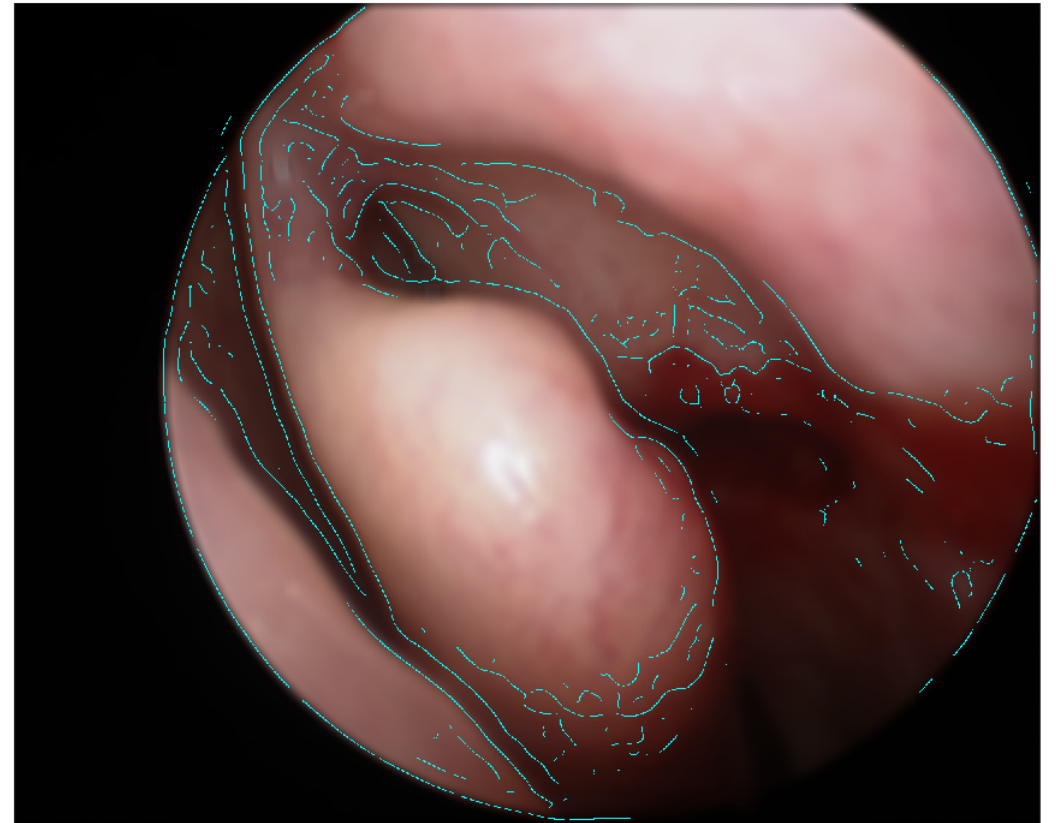
# Problems

- Although there is no extraneous texture in high intensity areas, low intensity, low contrast areas still exhibit artifacts.
- Missing edges occur along darker edges as the contrast isn't as detectable.
- Still need a method of removing artifacts from low contrast, low intensity pixel groups.
- Possible solution is to also analyze contrast in a radius around each pixel and filter out the pixels whose contrasts are low.

# Intensity Filter on Canny Edge Detection

- Considered using intensity filtering on canny edge detection results from before.
- Loses too much data at the same intensity threshold.
- At higher thresholds offered some improvement over texture filtering.

# Intensity Filter on Canny Edge Detection

- Considered using intensity filtering on canny edge detection results from before.
- Loses too much data at the same intensity threshold.
- At higher thresholds offered some improvement over texture filtering.