Virtual Simulation of Angiography and Perfusion MRI

Members of the team:

Gregg Hebert – 1657892 – NSM - Undergraduate
Mubashir Khan - 1521657 - NSM - Undergraduate
George Coll Rodriguez - 1529011 – NSM – Undergraduate

Introduction:

For this project, we constructed two videos of contrast agent running through a phantom of a heart. As the contrast agent passed through the phantom of the heart, we recorded the signal intensity of the contrast agent over time as a histogram. The two videos show the difference in how the contrast agent passes through a heart with normal perfusion, versus abnormal perfusion.

Aims:

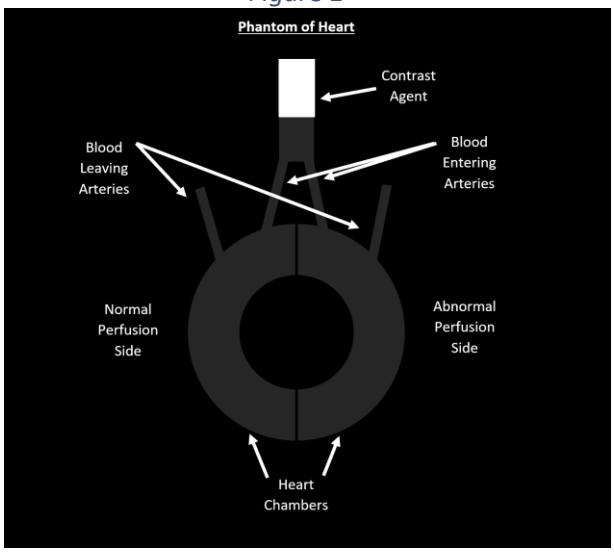
1. Create a distinct colored contrast agent. In real life angiography, the contrast agent usually appears black as is flows through the arteries and the background with appear gray/white. For our project, we went with a white agent and black background.

Our conclusions indicate that the contrast agent flows differently between the normal and abnormal perfusions. The normal perfusion histogram showed a bell curve of the signal intensity as it passed through the heart. The abnormal perfusion histogram had a sloped signal intensity as it passed through the heart. Another difference is the amount of time that passed for each perfusion. The contrast agent passed much slower the abnormal perfusion than the normal perfusion.

Methods:

To create the normal and abnormal perfusion videos, we used Microsoft PowerPoint. The phantom and contrast agent were created by using basic shapes that overlay each other. (The different areas of the phantom are labeled in *Figure 1* below.) With the phantom of our heart at the center of each slide, we simulated the contrast agent flowing through the heart by moving the shapes little-by-little, from one slide to another. The slides basically represent frames in a movie. Once we completed the all the slides, the PowerPoint was exported as an .mp4 file.

Figure 1

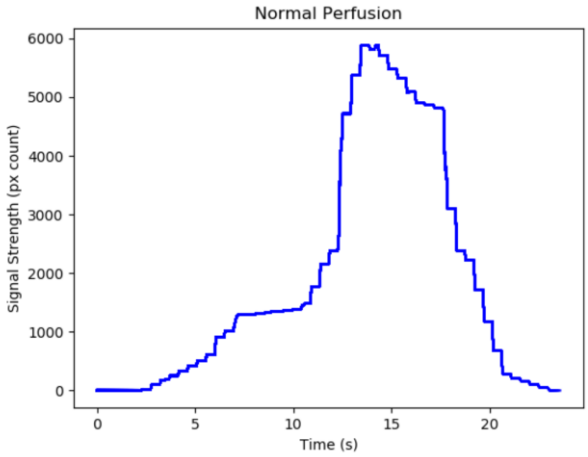


Python was used to read the videos and create the histograms of the signal intensities for both normal and abnormal perfusion. The library OpenCv has a function called 'VideoCapture' that can take in an .mp4 file as a variable. To capture the intensity of the contrast agent while the video is running, we need to get the pixel color of the contrast agent. Pixels are divided into 3 sections: Red, Green, and Blue. Each of these sections has a brightness intensity ranging from 0 to 255. So, if all 3 sections are 0, or [0, 0, 0], then we get the color black. If all the sections are 255, or [255, 255, 255], then we get the color white. Everything in between these two values represents some other color. However, since we are working with a grayscale video, we only need 1 section, or channel. With grayscale, 0 still represents black, 255 represents white, and everything else represents a variation of gray. For this project, since we are working with a grayscale image, we need a distinct color to represent the contrast agent (Aim 1).

With white as our choice of color for the contrast agent, we need to capture how many white pixels there are in each second of the video. We do this by using a while loop (for as long as the video is playing) and we append the count of white pixels to an array. Then, we plot the array, using the library MatPlotLib, to the y-axis of our histogram. The x-axis of our histogram represents the time in seconds, which is calculated by dividing the while loop iterations by the number of frames/second (which was set when exporting the video).

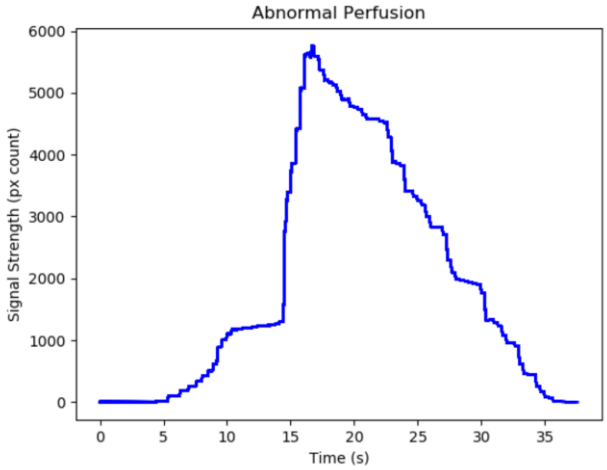
Results:

Normal Perfusion:



From these results, we can sort of see bell curve starting around the 10 second mark. As the contrast agent enters and fills the chamber of the heart, the intensity spikes because the of the amount of contrast agent (or pixels in this case). Once the contrast agent stops being injected, it quickly leaves the chamber and the signal intensity falls just as fast.

Abnormal Perfusion:



From these results, the intensity starts out like the normal perfusion, but the ending is a little different. Instead of the intensity dropping quickly as the contrast agent leaves, in slowly slopes down over a longer period of time as compared to the normal perfusion. This implies that the contrast agent is not leaving normally.

Conclusion:

In real life, abnormal perfusion can cause the contrast agent to flow slowly than normal, run backward, or even not flow at all (blockage). Since one our PowerPoint video was to represent an abnormal perfusion, from our results we see that the contrast agent does flow more slowly compared to the normal perfusion. Our results may not be highly accurate, but it does show a distinguishable difference between what is normal and what is not.

Bibliography:

Used as a reference while building project.
https://nrsyed.com/2018/02/08/real-time-video-histograms-with-opencv-and-python/

Used to understand OpenCV.
https://readthedocs.org/projects/opencv-python-tutroals/downloads/pdf/latest/

Code:

```
"""
Gregg Hebert – 1657892 – NSM - Undergraduate
Mubashir Khan - 1521657 - NSM - Undergraduate
George Coll Rodriguez - 1529011 – NSM – Undergraduate

Instrustions:
-  Make sure the two .mp4 files are in the same folder as this script.
-  Make sure all necessary libraries are installed.
-  Run the program.
-  Once the first video ends, it will close automatically.
-  Close the first histogram after you are done observing it by clicking the 'X'
in the top right corner.
-  The second set of video and histogram will persue automatically.
-  Once the second video ends, it will close automatically.
-  Close the second histogram after you are done observing it by clicking the 'X'
 in the top right corner.
-  Two .png files of the histograms will be added to the folder after closing the
 last histogram.

Note: press q at any time while the video is running to stop script.
"""
import numpy as np
import matplotlib.pyplot as plt
import cv2
import time
```

```
# parameters(the .mp4 video, number of frames in video (can be adjusted), type of
 perfusion(as string))
def getData(capture, numFrames, typePerfusion):
    i = 0
    x, y = [], []

    while True:
        (grabbed, frame) = capture.read()

        if not grabbed:
            break
        # Resize frame to width, if specified.
        if resizeWidth > 0:
            (height, width) = frame.shape[:2]
            resizeHeight = int(float(resizeWidth / width) * height)
            frame = cv2.resize(frame, (resizeWidth, resizeHeight),
                               interpolation=cv2.INTER_AREA)

        # Normalize histograms based on number of pixels per frame.
        numPixels = np.prod(frame.shape[:2])

        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        cv2.imshow(typePerfusion + ' Perfusion', gray)

        x.append(i/numFrames)

        # append the number of white pixels to array y
        y.append(np.count_nonzero(gray == 255))

        # plot in real-time with video
        ax.plot(x, y, color='b')
        fig.canvas.draw()

        i += 1

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    capture.release()
    cv2.destroyAllWindows()
    fig.savefig(typePerfusion+'.png')
    plt.show()


capture1 = cv2.VideoCapture('Normal_Perfusion.mp4')
```

```
color = 'gray'
typePerfusion = 'Normal'

# rough estimate of amount of frames/sec
numFrames = 15

# set width - can be adjusted fit your screen
resizeWidth = 700

# Label the plot axies and title
fig = plt.figure("Angiography Normal Perfusion Histrogram")
ax = fig.add_subplot(111)
ax.set_title('Normal Perfusion')
ax.set_xlabel('Time (s)')
ax.set_ylabel('Signal Strength (px count)')
fig.show()
getData(capture1, numFrames, typePerfusion)

capture2 = cv2.VideoCapture('Abnormal_Perfusion.mp4')

color = 'gray'
typePerfusion = 'Abnormal'
resizeWidth = 700

# Label the plot axies and title
fig = plt.figure("Angiography Abnormal Perfusion Histogram")
ax = fig.add_subplot(111)
ax.set_title('Abnormal Perfusion')
ax.set_xlabel('Time (s)')
ax.set_ylabel('Signal Strength (px count)')
fig.show()
getData(capture2, numFrames, typePerfusion)
```

Note: We did not do a complete interactive GUI because there are no necessary parameters that need to be changed for this project.