NGENUITY 3D Visualization Automation - Alpro

Team Name: Alpro
Group Members:

Abby Wysopal (Team Lead) - abbywysopal@ucsb.edu
Brent Luker (Scribe) - brentluker@ucsb.edu
Kiet Nguyen - knguyen104@ucsb.edu
Michael Hau - mdhau@ucsb.edu
Ryan Mitchell - ryan_mitchell@ucsb.edu

Introduction

Motivation and Background

Streamlining eye surgery through automation means that doctors will be able to bring improved vision to more patients and restore one of the most important human senses. With the prediction that “the world’s population will increase by about one-third” [1] and along with it, the demand for cataract surgery, optometrists will struggle to keep up with the demand. At the relatively low rate at which optometrists are trained, the demand will outpace the current solution. Automation of the NGENUITY 3D Visualization system will drastically cut down the amount of time medical professionals will need to spend on preparing the patient for surgery and in turn allow optometrists to help more patients.

Existing Solutions

The increase in cataract surgery demand is already causing great concern in the medical community and there are solutions currently being used to alleviate the problem. Beyond changes in administrative workflow, using lasers to make incisions increases consistency and accuracy while decreasing the amount of time to make the incisions, however even with advances in the actual surgical operation technology, systems using lasers will still benefit from decreasing the actual setup time through automation.

Core Components

Facial Detection

The facial detection will be used to both uniquely identify the patient and determine where precisely they are in the room so that the robot can prepare for surgery. The facial detection algorithm will be run on each frame of the video four times, once at each right angle of the frame, to detect the patient at any orientation. Additionally, for each frame, the algorithm will compare the current detected face, with any previously recorded faces to determine if it is the same face. If the eye coordinates and the orientation of the current face is similar to a previous face, the algorithm will update that face, rather than saving the current face as a new face. The patient will be determined as the face that has the highest probability of being a face, and has been in the most frames of the video. The coordinates of the patient’s eyes, and the orientation of their face will be saved and sent to the robot controller, so that the robot can be moved above the correct eye of the patient, and rotated to the desired orientation.

Eye Detection

The eye detection algorithm will use iris and pupil detection to detect the patient's eye. The eye detection algorithm uses the eye coordinates returned from the facial detection algorithm to move the robot camera into the relative area of the patient's eye then uses image processing techniques to perform iris and pupil detection. Once the iris and pupil are detected the eye detection interacts with the robot to make use of the existing focus and zoom functions on the camera to clarify the feed of the eye.

Simulation Environment

The simulation environment will act as an abstraction of the actual environment that the software will be used in. The simulation will include basic controls including: pan left and right to mimic changing x/y-coordinates and a zoom to mimic z-coordinates. The controls will be called by the facial detection algorithm to center and focus the camera onto the patient. The controls are implemented using HTTP requests to the local IP address of the camera. Each time controls need to be accessed, a token is granted to the process looking to access the controls; the token is then used indefinitely until desired controls have been called. The token is then released.

Goals

The primary goal of the project is to decrease the amount of time the surgeon needs to spend situating and configuring the camera onto the patient before surgery; this end goal can be broken down into several key steps:

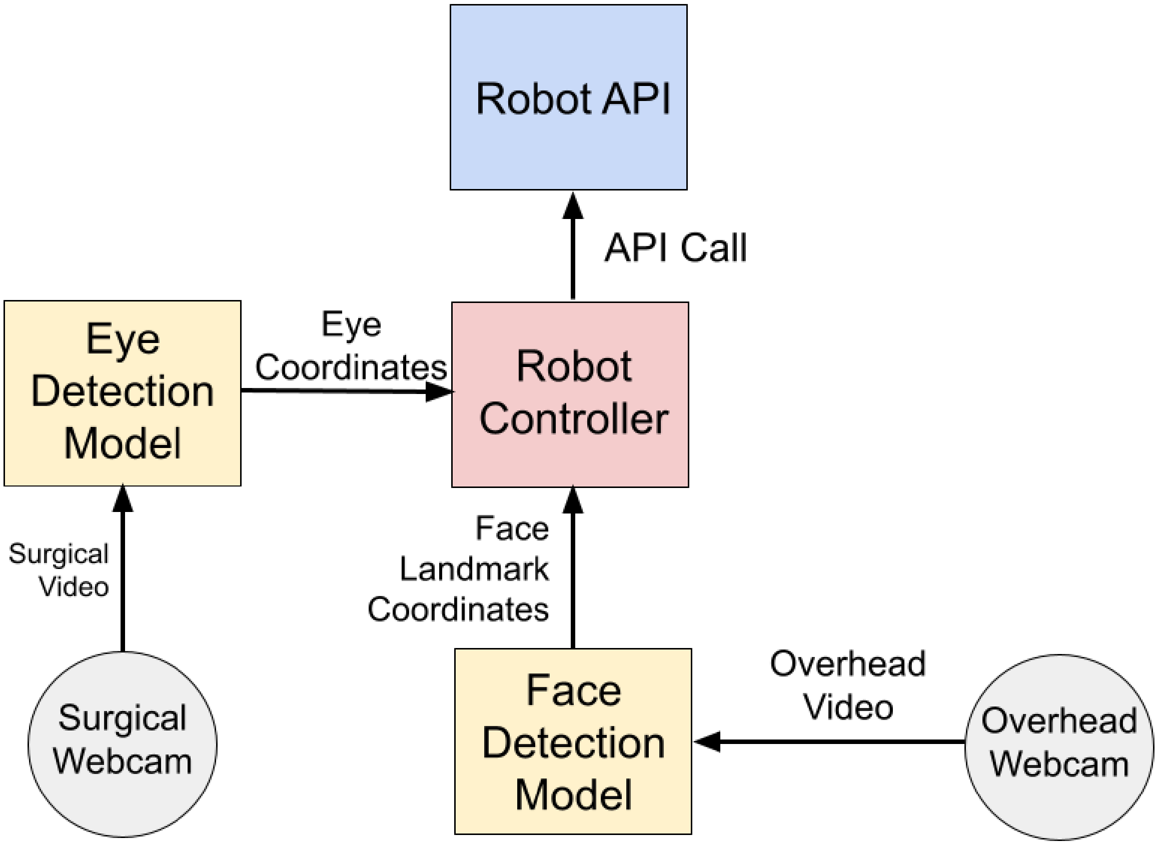
1. The robotic arm will automatically situate itself over the patient so that the camera on the arm will be able to detect the patient while ignoring other subjects in the room.
2. The camera on the robotic arm will zoom in and focus on the correct eye requiring surgery.
3. The overall experience for the operating team will be streamlined and user friendly while requiring almost no direct control.

Assumptions

Eye surgeries are expected to take place in a controlled environment with some variables but many invariants. The key invariants that the automation software will require include:

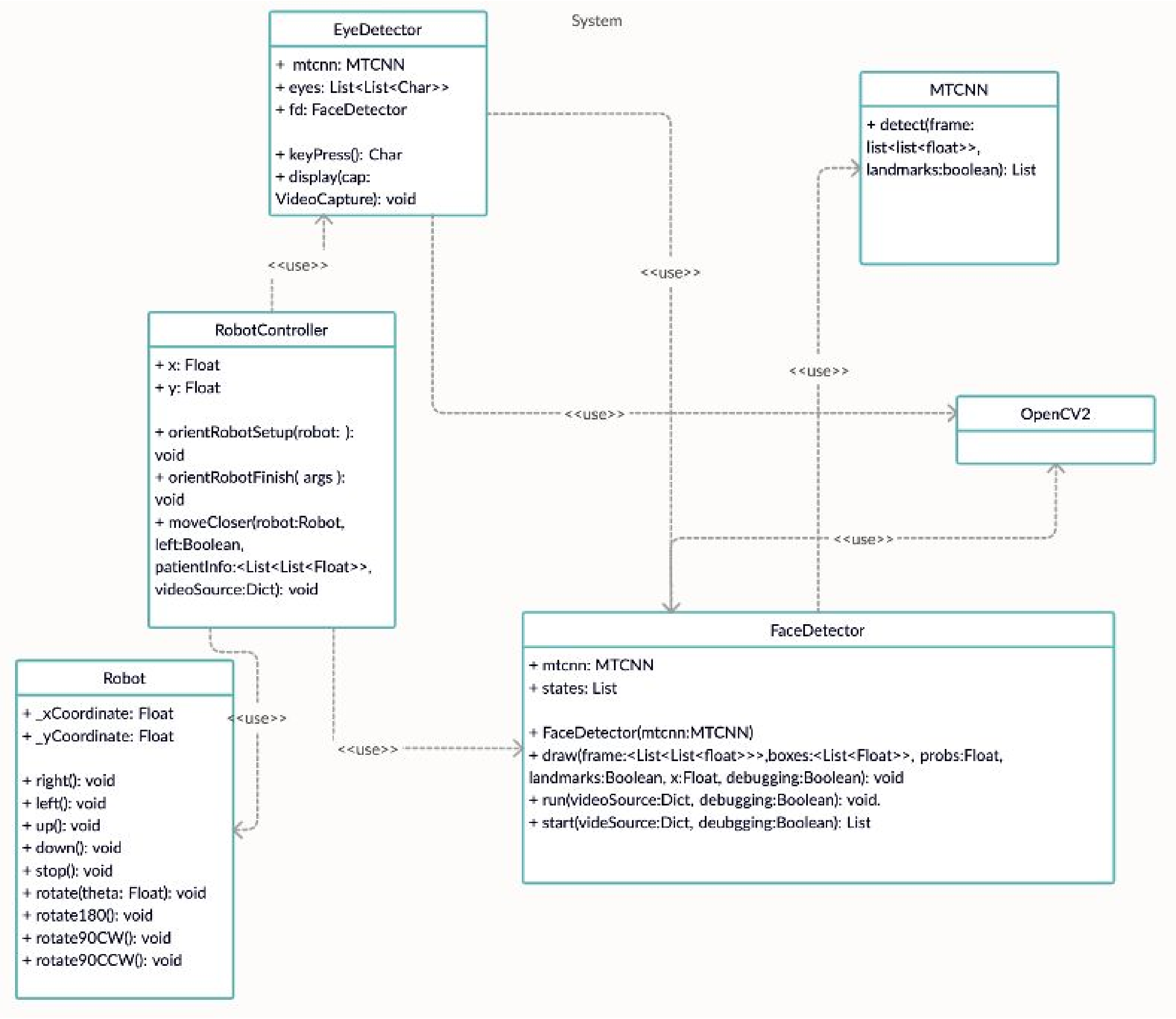
1. Only the patient will be facing upwards for the majority of the surgery, so that the bird's eye view camera will know which subject is the patient.
2. The lighting in the room will be adequate (adequate level of lighting is to be determined) for the camera to detect objects in the room.
3. Nothing will obstruct the patient's face during the facial detection phase.

Architecture Overview:

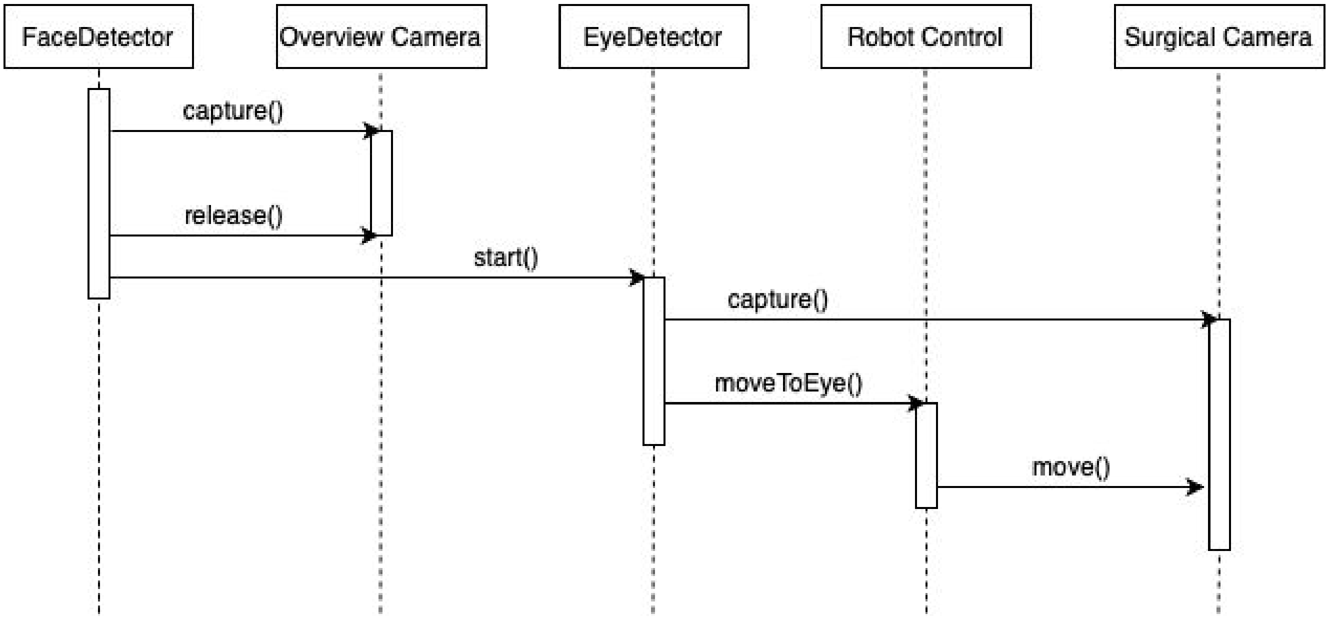


System Models:

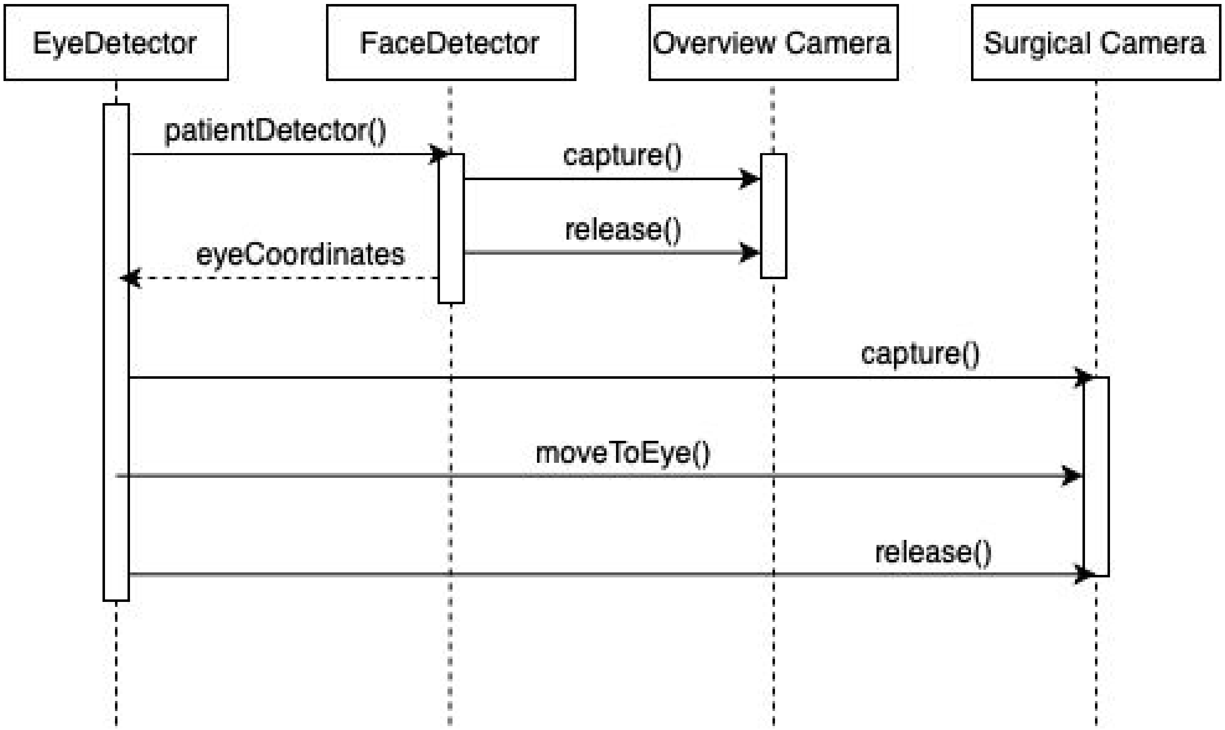
UML Design:


System
EyeDetector
+ mtcnn: MTCNN
+ eyes: List<List<Char>>
+ fd: FaceDetector
+ keyPress(): Char
+ display(cap: VideoCapture): void
MTCNN
+ detect(frame: list<list<float>>, landmarks:boolean): List
<<use>>
<<use>>
<<use>>
RobotController
+ x: Float
+ y: Float
+ orientRobotSetup(robot:): void
+ orientRobotFinish(args): void
+ moveCloser(robot:Robot, left:Boolean, patientInfo:<List<List<Float>>, videoSource:Dict): void
<<use>>
OpenCV2
<<use>>
Robot
+ _xCoordinate: Float
+ _yCoordinate: Float
+ right(): void
+ left(): void
+ up(): void
+ down(): void
+ stop(): void
+ rotate(theta: Float): void
+ rotate180(): void
+ rotate90CW(): void
+ rotate90CCW(): void
<<use>>
<<use>>
FaceDetector
+ mtcnn: MTCNN
+ states: List
+ FaceDetector(mtcnn:MTCNN)
+ draw(frame:<List<List<float>>>,boxes:<List<Float>>, probs:Float, landmarks:Boolean, x:Float, debugging:Boolean): void
+ run(videoSource:Dict, debugging:Boolean): void.
+ start(videSource:Dict, deubgging:Boolean): List

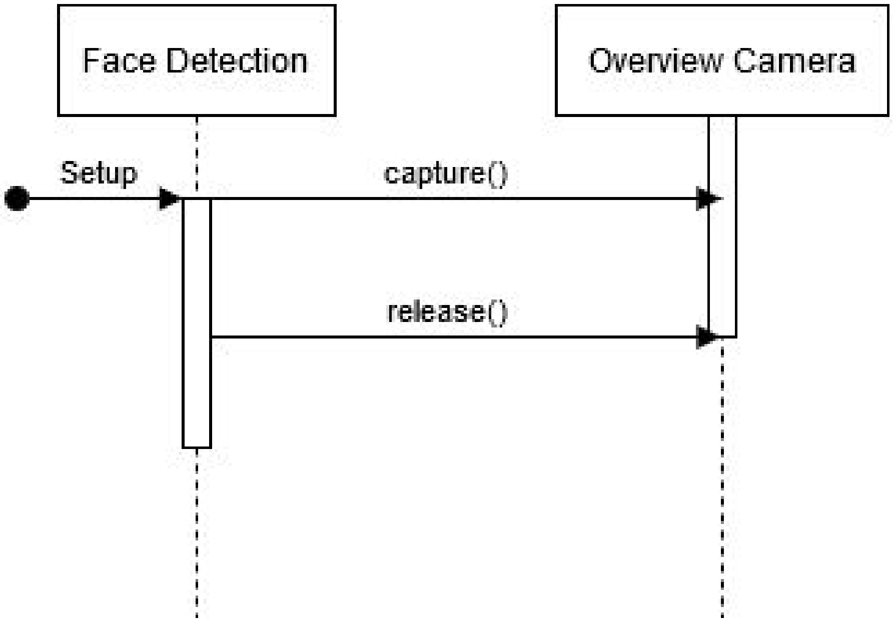
System Sequence:



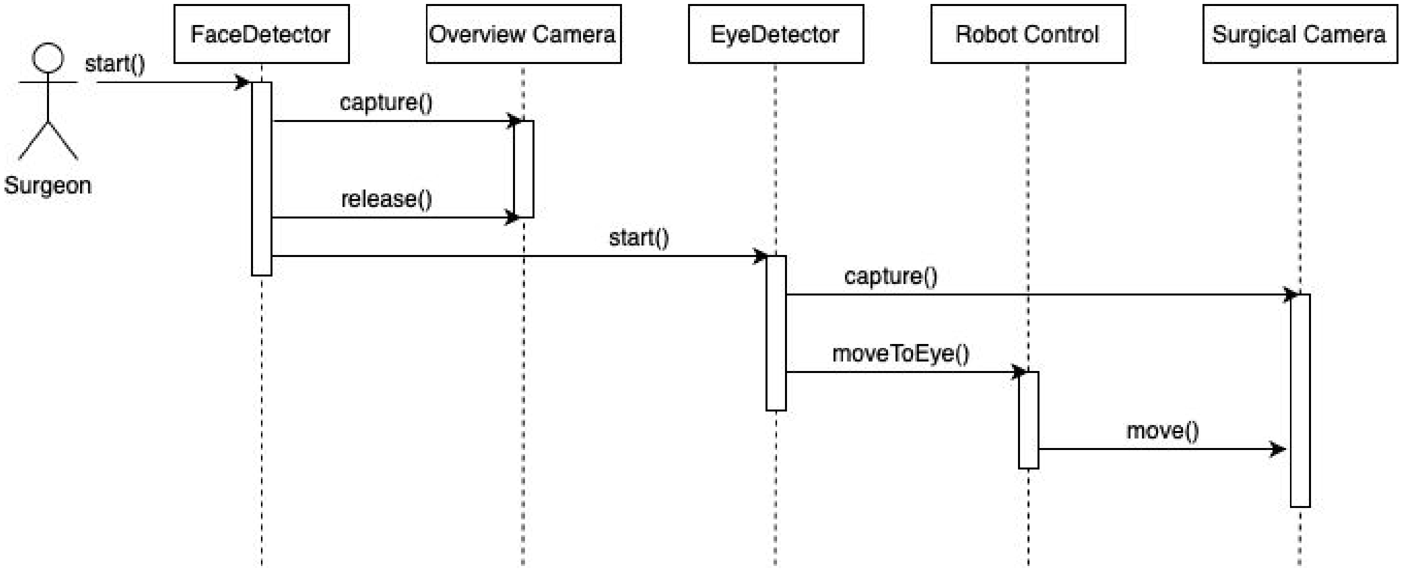
Eye Detection Sequence:



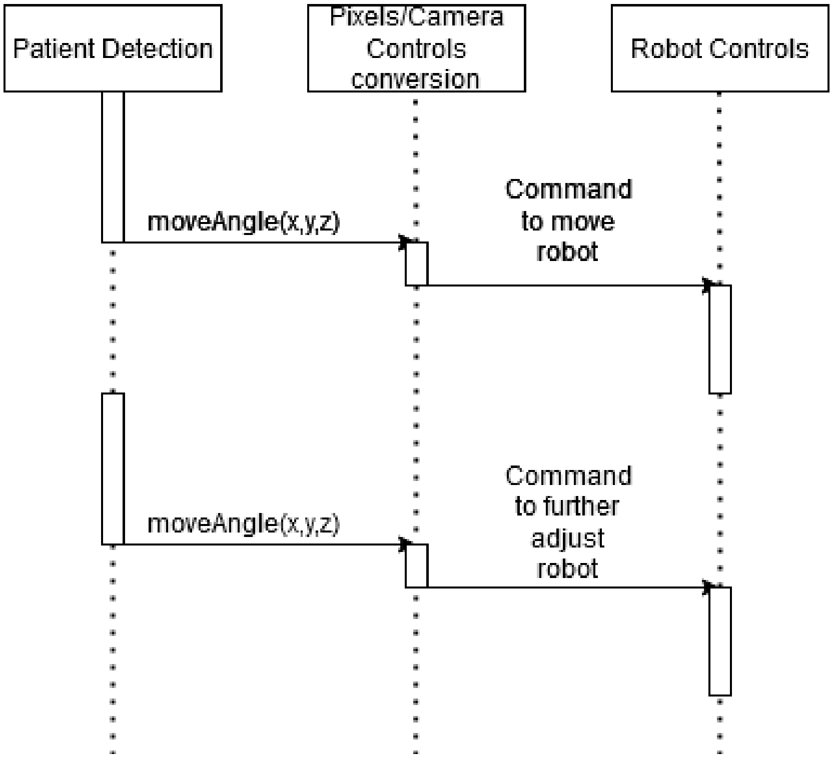
Face Detection Sequence:



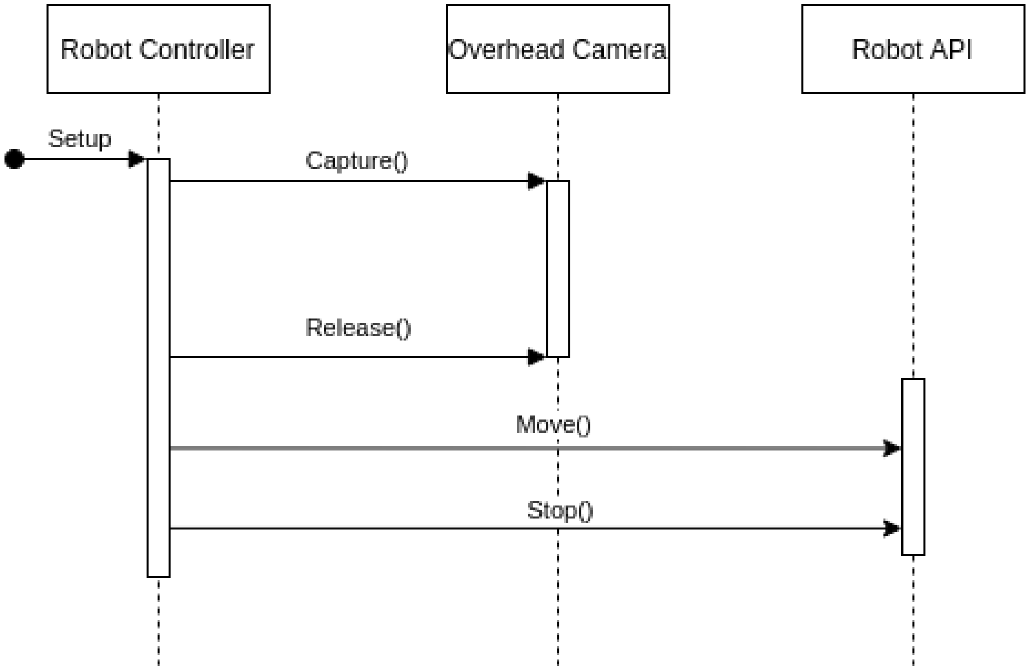
User Interaction Sequence:



Simulation Sequence:



Robot Control Sequence:



Requirements

1. **<u>As a programmer, I can use a face detection model to accuractly identify the patient's face from a webcam video feed</u>**

   Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/1**)

   a. Scenario 1: Patient is ready
   Given the patient is in the surgical chair, their face is accurately detected as the patient's face by the facial detection software.

   b. Scenario 2: No Patient in webcam
   Given there is no patient in the web camera, the facial detection software will determine which face has the highest probability of being the patient's face.

   Acceptance Criteria: Inputting a live video feed of the surgery room, the face detection model outputs which face is the patient's, and where the face is in the room.

2. **<u>As a programmer, I can find the eye coordinates from a face found in an image</u>**

   Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/5**)

   a. Scenario 1: One eye in image
   Given a face has been found in the image and there is only one eye in the image, then I will return the coordinates for the eye in the image

   b. Scenario 2: Two eyes in image
   Given a face has been found in the image and there is two eyes in the image, then I will return the coordinates for the two eyes in the image

   Acceptance Criteria: Inputting an image of a person's face, the eye detection model outputs the coordinates of the eyes.

3. **<u>As an Assistant Optometrist, I can press begin setup to initiate the setup process</u>**

   Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/8**)

   a. Scenario 1: Ready for set up
   Given the patient is in the surgical chair and ready for surgery and the Assistant Optometrist begins the setup process, then the set up process will initiate and a successful set up will occur.

   b. Scenario 2: Not ready for set up
   Given the patient is not ready for surgery and the Assistant Optometrist begins

the setup process, then the set up process will initiate and an error message will occur.

Acceptance Criteria: Initiating the start sequence, the app controller will throw an error message or continue setup according to the situation of the room.

4. **As an engineer, I can load my webcam feed into the simulation so that I can test my Machine Learning model**

   Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/10**)

   a. Scenario 1: Face Detection
      Given a video that a programmer wishes to test a face detection model on, the simulation can load and display the webcam video feed, as well as the results of the face detection model.

   b. Scenario 2: Eye Detection
      Given a video that a programmer wishes to test a face detection model on, the simulation can load and display the webcam video feed, and adjust the feed according to the signals of the eye detection optimizer.

   Acceptance Criteria: Inputting a webcam feed into the simulation, the machine learning models can run on the video feed and edit the video feed according to their respective goals.

5. **As an engineer, I can use image processing techniques to create a stabilized image**

   Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/9**)

   a. Scenario 1: Stable frame with moving objects
      Given a video or a live video recording with stable frames in which objects such as hands or the eyes are moving, the simulation will accurately determine when to stabilize the frames.

   b. Scenario 2: Shaky frame with moving objects
      Given a video or a live video recording with shaky frames, the simulation will attempt to stabilize the frames so that the images will look better for human vision without rendering illusions.

   Acceptance Criteria: Inputting a live video feed of the patient's eye, the camera will remain stabilized on the eye as it moves around.

6. **As an engineer, I want to create a dataset to classify eye images**

   Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/7**)

   a. Scenario 1: Ideal image
      Given an image is the ideal target image, it will be classified in the dataset as a good image

   b. Scenario 2: Bad image
      Given an image is not the ideal target image, it will be classified in the dataset as a bad image

   Acceptance Criteria: Inputting a set of eye images, the dataset will be correctly stored and labeled in the database.

7. **As an engineer, I can integrate eye detection software with the robot arm to move the surgical camera**

   Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/5)**

   a. Scenario 1: Move Closer to Eye
      Given the signal from the eye detection software, the corresponding robot api function is called and the robot moves closer to the eye.

   b. Scenario 2: Move Further from Eye
      Given the signal from the eye detection software, the corresponding robot api function is called and the robot moves further from the eye.

   c. Scenario 3: Move West of Eye
      Given the signal from the eye detection software, the corresponding robot api function is called and the robot moves west of the eye.

   d. Scenario 4: Move East of Eye
      Given the signal from the eye detection software, the corresponding robot api function is called and the robot moves east of the eye.

   e. Scenario 5: Move North of Eye
      Given the signal from the eye detection software, the corresponding robot api function is called and the robot moves north of the eye.

   f. Scenario 6: Move South of Eye
      Given the signal from the eye detection software, the corresponding robot api function is called and the robot moves south of the eye.

Acceptance Criteria: Inputting the eye detection output into the robot api, the robot moves in the correct direction depending on the output of the eye detection model.

8. **As a software engineer, I can access live webcam stream to create a simulation**

   Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/4**)

   a. Scenario 1: Webcam on
      Given the web camera is on, I can access the live webcam stream in my program

   b. Scenario 2: Webcam off
      Given the web camera is off, I will generate an error message

Acceptance Criteria: The livestream will operate on a wifi network and be accessible through a very nice GUI.

9. **As a software engineer, I can zoom in on live webcam stream to simulate surgical camera z-axis movement**

   Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/3**)

   a. Scenario 1: Closer to the Eye
      Given the live webcam stream in the simulation, I can call a function to zoom into the video and simulate robotic Z-axis movement towards the eye.

   b. Scenario 2: Further from the Eye
      Given the live webcam stream in the simulation, I can call a function to zoom out of the video and simulate robotic Z-axis movement away from the eye.

Acceptance Criteria: With a live webcam streaming in the simulation, the video stream is zoomed in or out depending on what function is called.

10. **As a software engineer, I can move around on live webcam stream x axis to simulate surgical camera moving east and west**

    Github links (issues): (**https://github.com/mycoal99/AlconCapstone/issues/2**)

    a. Scenario 1: Move stream east
       If there is a signal to move the stream to the east, the camera’s view should be shifted accordingly.

b. Scenario 2: Move stream west
   If there is a signal to move the stream to the east, the camera's view should be shifted accordingly.

Acceptance Criteria: With a live webcam streaming in the simulation, the video stream is moving left or right depending on what function is called.

11. **<u>As a software engineer, I can access the NGENUITY API through json sockets</u>**
    Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/13

    a. Scenario 1: Integration of system control algorithms
       With the system control algorithms, I can control the NGENUITY system using their NGENUITY API using JSON requests.

    b. Scenario 2: Collecting robot metadata
       Using simple JSON requests, I can grab the system's robotic metadata.

    Acceptance Criteria: With the system control algorithms, I can control the NGENUITY system and grab the system's robotic metadata.

12. **<u>As a software engineer, I can move around on live webcam stream y axis to simulate surgical camera moving north and south</u>**
    Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/14

    a. Scenario 1: Move stream north
       If there is a signal to move the stream to the north, the camera's view should be shifted accordingly.

    b. Scenario 2: Move stream south
       If there is a signal to move the stream to the south, the camera's view should be shifted accordingly.

    Acceptance Criteria: With a live webcam streaming in the simulation, the video stream is moving up or down depending on what function is called.

13. **<u>As a software engineer, I can access the live web camera stream in a birdseye view</u>**
    Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/15

    a. Scenario 1: Collect frame information for processing
       Frames from the camera stream can be taken so that I can use them as data for processing and training.

b. Scenario 2: Verification and calibration of operating environment
   Before using a testing environment, I can manually verify that the operating camera and subject are in frame.

Acceptance Criteria: Web camera is streaming in a birdsview eye.

14. **<u>As a software engineer, I can access the live web camera stream in a specific view to simulate surgical camera</u>**
    Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/16
    a. Scenario 1: Surgical Camera View
       If the software is running patient eye detection, the video stream will have a surgical camera view

    b. Scenario 2: Birdseye View
       If the software is running patient detection, the video stream will have a birds eye view

    Acceptance Criteria: Simulation changes views from birdseye view to surgical camera view.

15. **<u>As a software engineer, I can develop an algorithm to correctly place the camera on patients eye</u>**
    Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/17
    a. Scenario 1: Iris detected
       If an iris is detected, I can zoom the camera closer to the iris until the detected iris 90% percent of the displayed image.

    b. Scenario 2: No Iris detected
       If no iris is detected, I can use the robot arm to scan until iris is detected and perform patient detection to find coordinates of patient eyes.

    Acceptance Criteria: The algorithm will detect the presence of an eye if and only if there is an eye and then zoom in on the iris, or else the robot arm is moved until there is an eye detected.

16. **<u>As the robot controller, I can identify the robot via qr code, and store its orientation and position</u>**
    Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/18
    a. Scenario 1: QR code not askew

If the QR code is not askew in the image, the overhead camera reads the code, stores a multiple of pi/2 as its orientation, and the position of its center in the robot class.

b. Scenario 2: QR code askew
If the QR code is askew in the image, the overhead camera reads the code, and correctly stores its orientation and the position of its center in the robot class.

c. Scenario 3: no QR code present
If the QR code is not present in the overhead camera feed, then the robot controller exits and sends an error message that the robot is not in the frame.

Acceptance Criteria: The QR code is recognized by the robot controller and the controller stores the position and orientation data if the code is present, otherwise it returns an error message.

17. **As a software engineer, I can store the eye coordinates detected from patient recognition software**

Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/19

a. Scenario 1: Both eyes are in frame
Patient recognition software returns the coordinates of the eyes in the patient object.
b. Scenario 2: One eye is in frame
Returns an error message saying there is no patient in frame
c. Scenario 3: No eyes are in frame
Returns an error message saying there is no patient in frame

Acceptance Criteria: Patient recognition software returns the coordinates of the eyes if both eyes are in the frame, if not, the software returns an error message.

18. **As the robot controller I can access the robot api, the patient data, and the current robot data in order to move the robot to the patient**

Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/20

a. Scenario 1: Wireless connection to Overhead Camera
With a wireless connection to the Overhead Camera, the patient's face and the robot's QR code can be identified with low latency and communicated to the robot controller.

b. Scenario 2: Wired connection to Overhead Camera

With a wired connection to the Overhead Camera, the patient's face and the robot's QR code can be identified with low latency and communicated to the robot controller.

Acceptance Criteria: Regardless of the connection type the robot controller has with the overhead camera, the robot controller can use the overhead camera to run patient recognition and robot recognition with low latency.

19. **As the patient eye detector, I can detect the patient's iris**

    Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/21

    a. Scenario 1: Eye in image
    If there is an eye in the image, I can use image processing to detect the patient's iris.

    b. Scenario 2: No Eye in image
    If there is no eye in the image, I can move the robot arm to scan until there is an eye in the image.

    Acceptance Criteria: The patient eye detector will detect the presence of an eye if and only if there is an eye or else the robot arm is moved until there is an eye detected.

20. **As a webcam controller, I can authenticate and access the webcam**

    Github links (issues): https://github.com/mycoal99/AlconCapstone/issues/22

    a. Scenario 1: Using the login credentials, I can retrieve an authenticated token to then interface with the camera.

    b. Scenario 2: Once authentication has been completed, I can retrieve the payload of camera controls to recreate the commands in Python.

    Acceptance Criteria: The stream controller can access all functionality of the camera as a user would be able to, using requests.

Appendix

Tech Stack: Python, PyTorch, NGENUITY 3D Robot API, OpenCV, Javascript, Google Colaboratory, Google Cloud