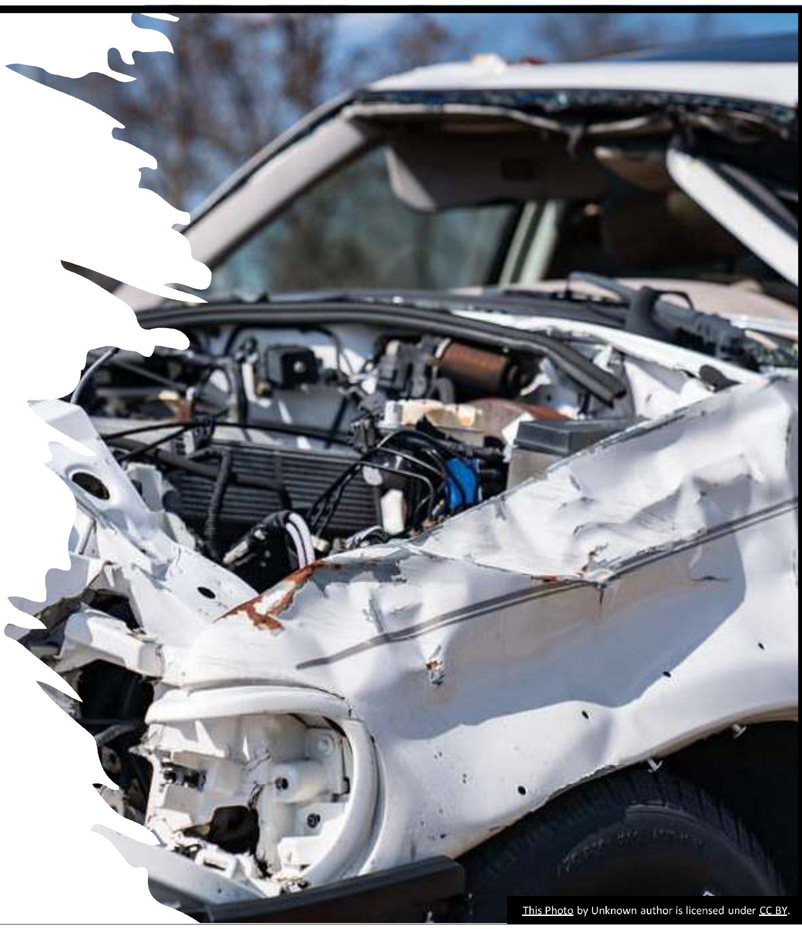
AmEyeDrunk? The Future of Intoxication Detection

Introduction

Intoxicated driving poses a critical public health issue in the United States. According to the CDC (Centers for Disease Control and Prevention),

- **32 people in the United States are killed every day in crashes involving an alcohol-impaired driver**
- **That is one death every 45 minutes**

The consumption of alcohol and drugs can significantly impair one's ability to accurately judge their level of impairment, leading to tragic consequences on the roadways.



Preliminary Research

Drinking alcohol causes the pupils to dilate or expand.

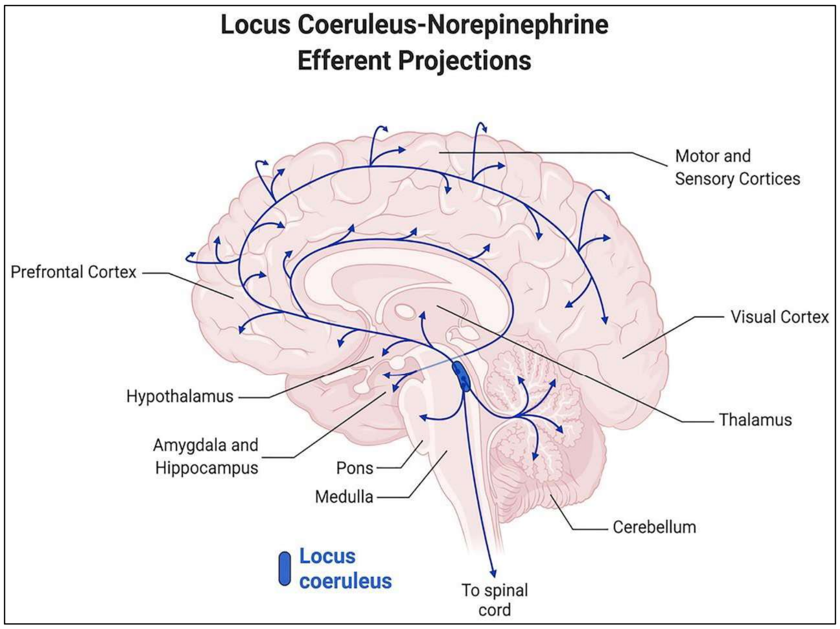
The muscles in the iris, which controls the size of the pupil, become relaxed because of alcohol on the body.

In the state of Washington, the legal limit for blood alcohol content (BAC) while driving is 0.08%.

Pupillometry is a technique used to measure changes in pupil size.

The locus coeruleus is a brain structure that controls pupil dilation and constriction.

Alcohol diminishes norepinephrine release in the locus coeruleus, which can affect pupil size and other physiological processes in the body.



Research Questions

1. **Can a mathematical model be developed to accurately predict drug/alcohol intoxication based on changes in pupil size and reactivity?**
2. How well does the model **integrate into an app to detect drug/alcohol intoxication** compared to traditional methods (e.g., breathalyzers, blood tests)
3. How can the app be designed to protect the privacy and rights of individuals who are being tested for drug/alcohol intoxication?
4. How can the app be user-friendly and easy for law enforcement and other users?
5. **How can the app be adapted for other scenarios, such as medical imaging and security systems?**

Hypothesis

If I can develop an app that utilizes computer vision algorithms to analyze pupil size and reactivity, it will accurately predict drug/alcohol intoxication with a sensitivity and specificity of at least 90% compared to traditional methods such as breathalyzers and blood tests.

This hypothesis states that the app, which uses computer vision algorithms to analyze pupil size and reactivity, can accurately predict drug/alcohol intoxication with high accuracy (90% or higher) compared to traditional methods. It also sets a specific benchmark for the app's performance in terms of sensitivity and specificity, which measures how well the app can correctly identify individuals under the influence and those not.

Procedure

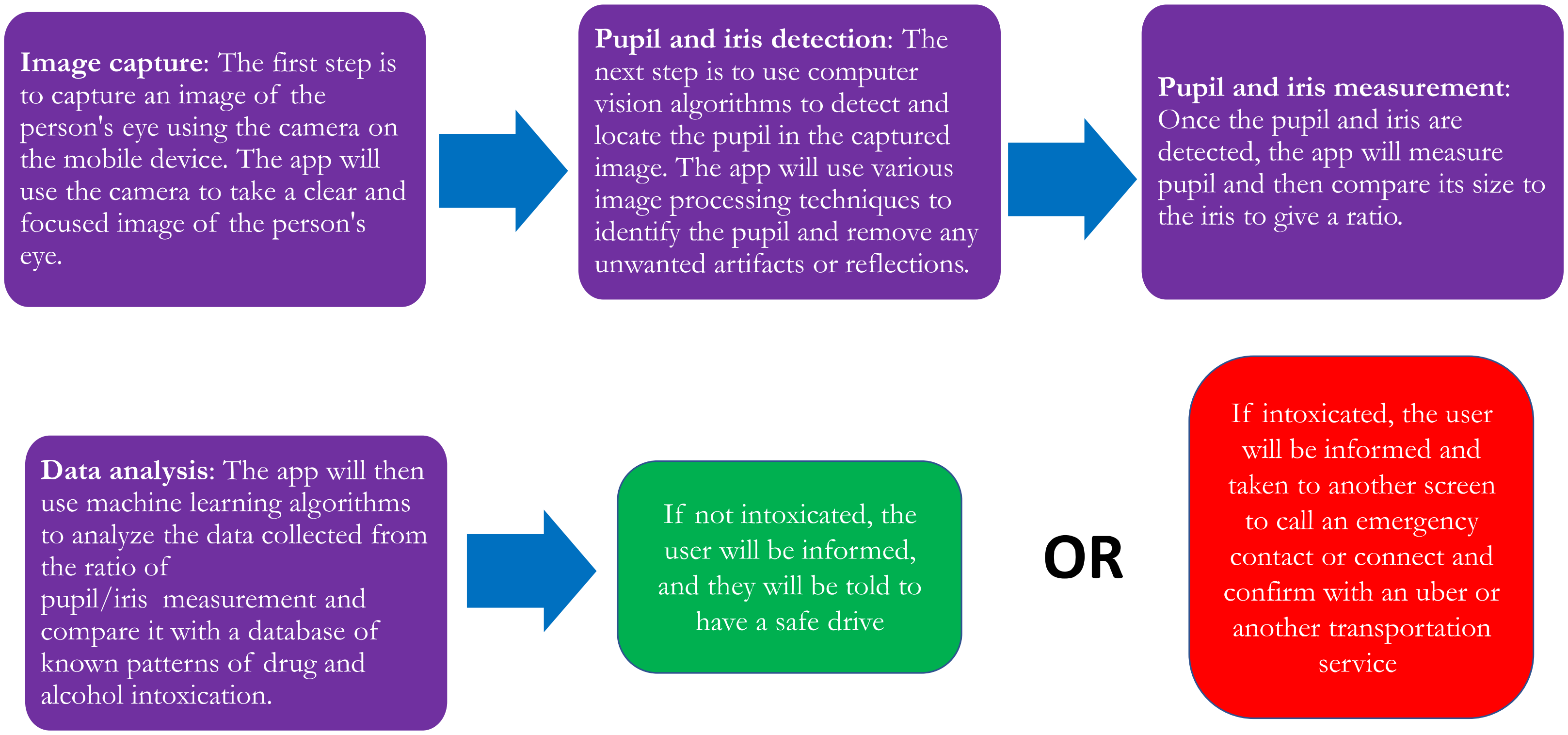


Image capture: The first step is to capture an image of the person's eye using the camera on the mobile device. The app will use the camera to take a clear and focused image of the person's eye.

Pupil and iris detection: The next step is to use computer vision algorithms to detect and locate the pupil in the captured image. The app will use various image processing techniques to identify the pupil and remove any unwanted artifacts or reflections.

Pupil and iris measurement: Once the pupil and iris are detected, the app will measure pupil and then compare its size to the iris to give a ratio.

Data analysis: The app will then use machine learning algorithms to analyze the data collected from the ratio of pupil/iris measurement and compare it with a database of known patterns of drug and alcohol intoxication.

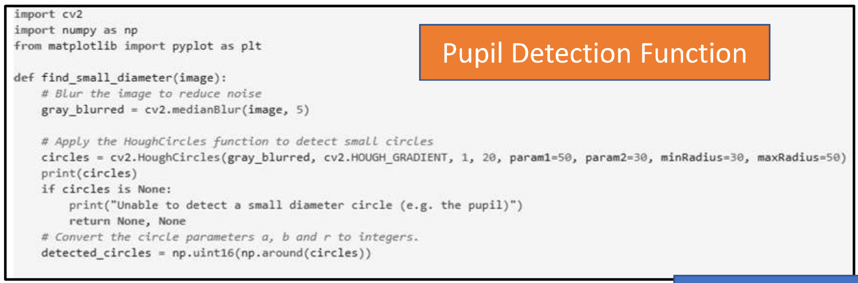
If not intoxicated, the user will be informed, and they will be told to have a safe drive

OR

If intoxicated, the user will be informed and taken to another screen to call an emergency contact or connect and confirm with an uber or another transportation service

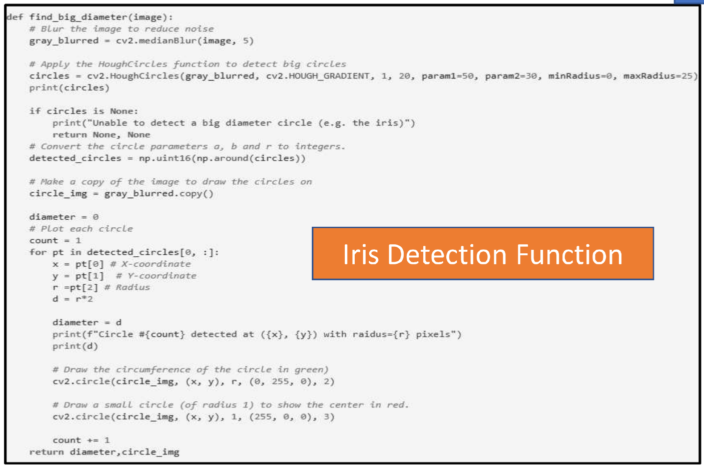
Detection Program

Pupil Detection Function



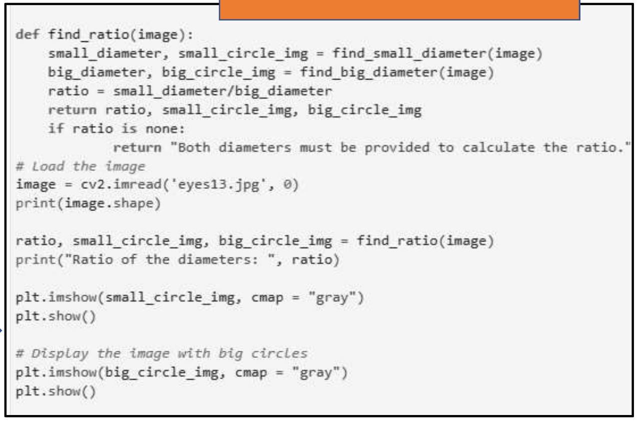
```python
import cv2
import numpy as np
from matplotlib import pyplot as plt

def find_small_diameter(image):
    # Blur the image to reduce noise
    gray_blurred = cv2.medianBlur(image, 5)

    # Apply the HoughCircles function to detect small circles
    circles = cv2.HoughCircles(gray_blurred, cv2.HOUGH_GRADIENT, 1, 20, param1=50, param2=30, minRadius=30, maxRadius=50)
    print(circles)
    if circles is None:
        print("Unable to detect a small diameter circle (e.g. the pupil)")
        return None, None
    # Convert the circle parameters a, b and r to integers.
    detected_circles = np.uint16(np.around(circles))
```

Iris Detection Function



```python
def find_big_diameter(image):
    # Blur the image to reduce noise
    gray_blurred = cv2.medianBlur(image, 5)

    # Apply the HoughCircles function to detect big circles
    circles = cv2.HoughCircles(gray_blurred, cv2.HOUGH_GRADIENT, 1, 20, param1=50, param2=30, minRadius=0, maxRadius=25)
    print(circles)

    if circles is None:
        print("Unable to detect a big diameter circle (e.g. the iris)")
        return None, None
    # Convert the circle parameters a, b and r to integers.
    detected_circles = np.uint16(np.around(circles))

    # Make a copy of the image to draw the circles on
    circle_img = gray_blurred.copy()

    diameter = 0
    # Plot each circle
    count = 1
    for pt in detected_circles[0, :]:
        x = pt[0] # X-coordinate
        y = pt[1]  # Y-coordinate
        r =pt[2] # Radius
        d = r*2

        diameter = d
        print(f"Circle #{count} detected at ({x}, {y}) with raidus={r} pixels")
        print(d)

        # Draw the circumference of the circle in green)
        cv2.circle(circle_img, (x, y), r, (0, 255, 0), 2)

        # Draw a small circle (of radius 1) to show the center in red.
        cv2.circle(circle_img, (x, y), 1, (255, 0, 0), 3)

        count += 1
    return diameter,circle_img
```

Ratio Calculation Function



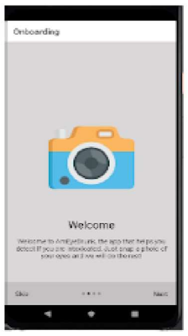
```python
def find_ratio(image):
    small_diameter, small_circle_img = find_small_diameter(image)
    big_diameter, big_circle_img = find_big_diameter(image)
    ratio = small_diameter/big_diameter
    return ratio, small_circle_img, big_circle_img
    if ratio is none:
            return "Both diameters must be provided to calculate the ratio."
# Load the image
image = cv2.imread('eyes13.jpg', 0)
print(image.shape)

ratio, small_circle_img, big_circle_img = find_ratio(image)
print("Ratio of the diameters: ", ratio)

plt.imshow(small_circle_img, cmap = "gray")
plt.show()

# Display the image with big circles
plt.imshow(big_circle_img, cmap = "gray")
plt.show()
```

find_small_diameter(image): This function takes an image as input and applies a median filter to reduce noise. It then applies HoughCircles to detect small circles (e.g. the pupil) within a specified radius range. It returns the diameter of the detected circle and an image with the detected circle drawn on it.

find_big_diameter(image): This function takes an image as input and applies a median filter to reduce noise. It then applies HoughCircles to detect larger circles (e.g. the iris) within a specified radius range. It returns the diameter of the detected circle and an image with the detected circle drawn on it.

find_ratio(image): This function takes an image as input and calls the **find_pupil_diameter** and **find_iris_diameter** functions to detect the small and big circles respectively. It then calculates the ratio of the diameters of the detected circles and returns the ratio along with images of the detected small and big circles.

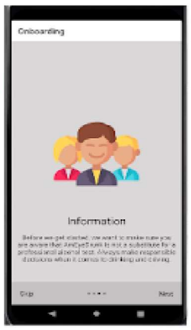
App Development


Onboarding

Welcome

Welcome to AmEyeDrunk, the app that helps you detect if you are intoxicated. Just snap a photo of your eyes and we will do the rest.

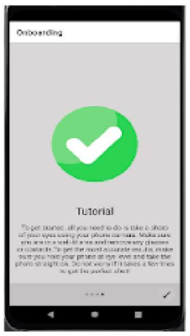
Skip Next


Onboarding

Information

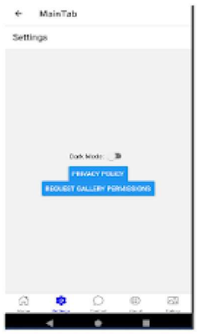
Before we get started, we want to make sure you are aware that AmEyeDrunk is not a substitute for a professional alcohol test. Always make responsible decisions when it comes to drinking and driving.

Skip Next


Onboarding

Tutorial

To get started, all you need to do is take a photo of your eyes using your phone camera. Make sure you are in a well-lit area and remove any glasses or contacts. To get the most accurate results, make sure you hold your phone at eye level and take the photo straight on. Do not worry if it takes a few tries to get the perfect shot!

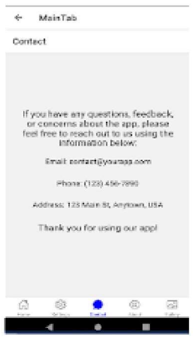

MainTab

Settings

Dark Mode:

PRIVACY POLICY

REQUEST GALLERY PERMISSIONS


MainTab

Contact

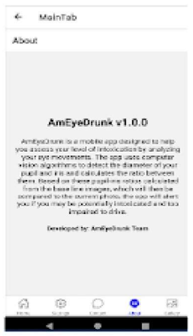
If you have any questions, feedback, or concerns about the app, please feel free to reach out to us using the information below:

Email: contact@yourapp.com

Phone: (123) 456-7890

Address: 123 Main St, Anytown, USA

Thank you for using our app!

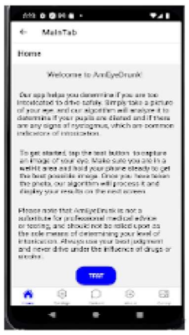

MainTab

About

AmEyeDrunk v1.0.0

AmEyeDrunk is a mobile app designed to help you assess your level of intoxication by analyzing your eye movements. The app uses computer vision algorithms to detect the diameter of your pupil and iris and calculates the ratio between them. Based on these pupil-iris ratios calculated from the base line images, which will then be compared to the current photo, the app will alert you if you may be potentially intoxicated and too impaired to drive.

Developed by: AmEyeDrunk Team


MainTab

Home

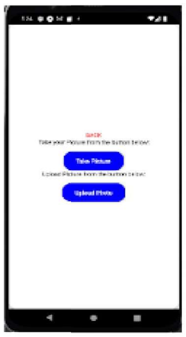
Welcome to AmEyeDrunk!

Our app helps you determine if you are too intoxicated to drive safely. Simply take a picture of your eye, and our algorithm will analyze it to determine if your pupils are dilated and if there are any signs of nystagmus, which are common indicators of intoxication.

To get started, tap the test button to capture an image of your eye. Make sure you are in a well-lit area and hold your phone steady to get the best possible image. Once you have taken the photo, our algorithm will process it and display your results on the next screen.

Please note that AmEyeDrunk is not a substitute for professional medical advice or testing, and should not be relied upon as the sole means of determining your level of intoxication. Always use your best judgment and never drive under the influence of drugs or alcohol.

TEST

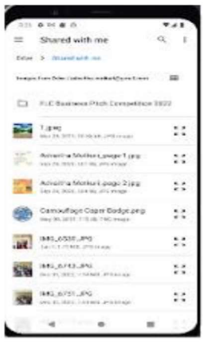

[illegible]

Take your Picture from the button below:

Take Picture

Upload Picture from the button below:

Upload Photo


Shared with me

Drive > Shared with me

FLC Business Pitch Competition 2022

1.jpeg

Advaitha Mothuri_page 1.jpg

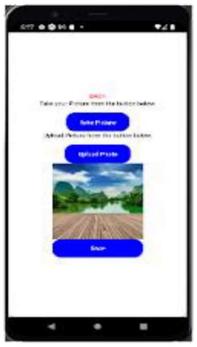
Advaitha Mothuri_page 2.jpg

Camouflage Copter Badge.png

IMG_6530.JPG

IMG_6743.JPG

IMG_6751.JPG


[illegible]

Take your Picture from the button below:

Take Picture

Upload Picture from the button below:

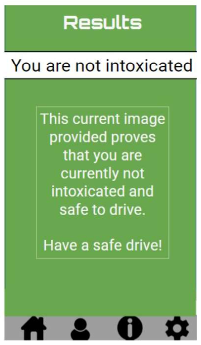
Upload Photo

Save

MainTab

Gallery

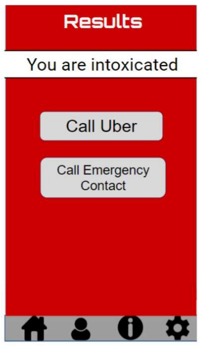
Loading image...


Results

You are not intoxicated

This current image provided proves that you are currently not intoxicated and safe to drive.

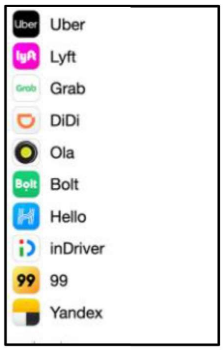
Have a safe drive!


Results

You are intoxicated

Call Uber

Call Emergency Contact


Uber

Lyft

Grab

DiDi

Ola

Bolt

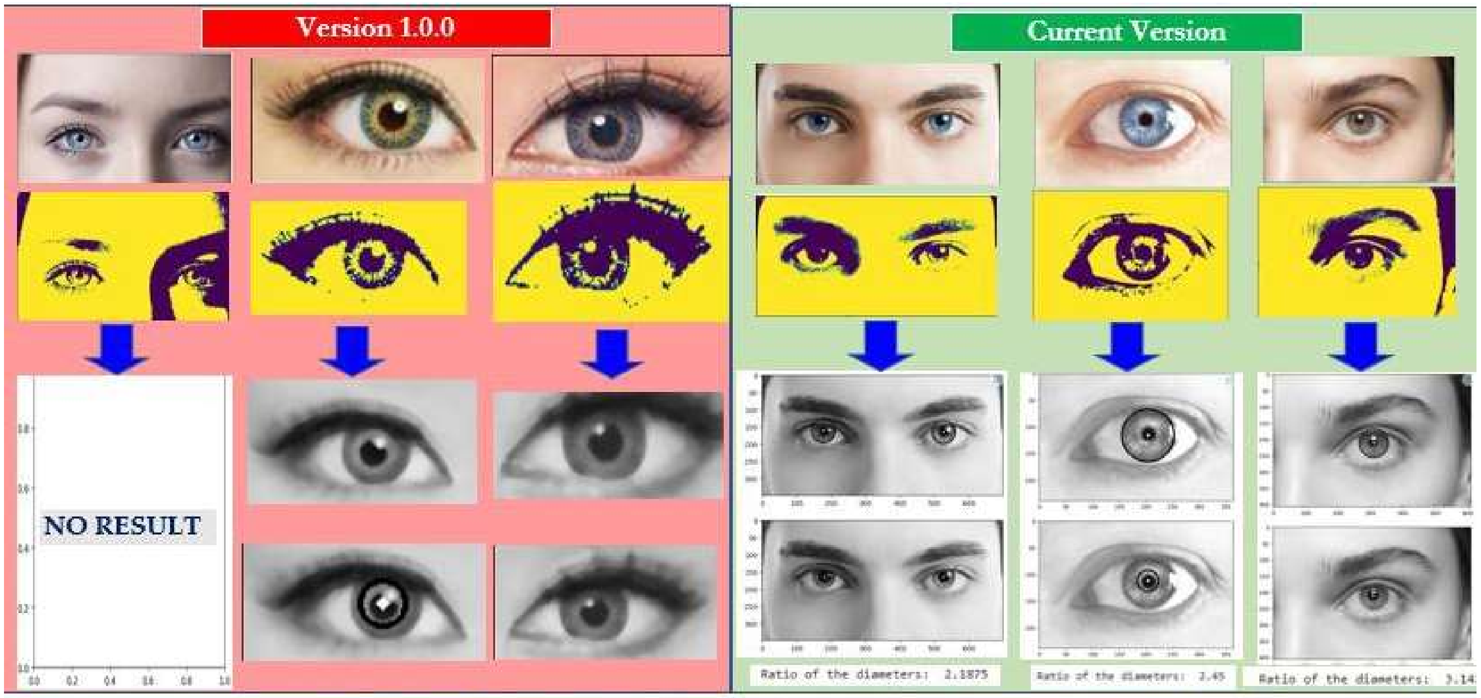
Hello

inDriver

99

Yandex

Results


Version 1.0.0
Current Version
NO RESULT
Ratio of the diameters: 2.1875
Ratio of the diameters: 2.45
Ratio of the diameters: 3.14

The original results had a success rate of about 5% because there was a lot of background noise in the images causing the algorithm not focusing but after further optimization and improvement such as adding the gray, blur filter to reduce the noise, we are now at a 95% success rate with almost all of the eyes giving us a pupil to iris ratio to detect the intoxication in a person. There is an 80% chance that the person is intoxicated if the ratio goes up by a whole number, then the algorithm will look at the redness of the eye and also reaction time of the pupil compared against the iris.

Conclusion

In conclusion, the **app accurately detects the diameters of the iris and pupil and calculates their ratios.** The app aims to provide **a tool** for individuals to assess their level of **intoxication** based on a ratio of the diameter of the small and big circles in their eyes. The app includes an onboarding process, a home screen with a test button, a privacy policy, a contact screen, and a settings screen for customization options. In the future, clinically testing this app results to compare with the FDA-approved pupillometer (a medical device intended to measure by reflected light the pupil size) **Overall, the app aims to promote safety by providing individuals with a quick and easy way to assess their level of intoxication and make informed decisions**.

Bibliography

Acknowledgments: Mr. Erol Cromwell (Mentor). Dr. Jarod Christiansen (Optometrist). Dr. Jagan Bontha (Advisor), Krystal Bass (Advisor)

"Circle Detection Using OpenCV Python." GeeksforGeeks, 8 July 2019, https://www.geeksforgeeks.org/circle-detection-using-opencv-python/.

Fletcher, Jenna. "What to Know about the Effects of Alcohol on the Eyes." Medical News Today, 21 Dec. 2022, https://www.medicalnewstoday.com/articles/alcohol-eyes#summary.

"Impaired Driving." Washington Traffic Safety Commission, 23 June 2014, https://wtsc.wa.gov/programs-priorities/impaired-driving/.

React Native · Learn Once, Write Anywhere. https://reactnative.dev/. Accessed 3 Feb. 2023.

Strauch, Christoph, and Marnix Naber. "Irissometry: Effects of Pupil Size on Iris Elasticity Measured With Video-Based Feature Tracking." Investigative Ophthalmology & Visual Science, vol. 63, no. 2, Feb. 2022, doi:10.1167/iovs.63.2.20.

"What Is a Mockup? (+How to Create a Mockup in 2022)." Clique Studios, 5 May 2022, https://cliquestudios.com/mockups/.

The programming languages used were Python, Javascript, CSS, and HTML. The interfaces used were Jupyter notebook with an Anaconda interface and React Native Studio with Visual Studio Code running on Android Studio and the **OpenCV library**.