DROWSINESS DETECTION

OUR TEAM

Nur Fazreen Binti Abdul Hafiz

Lai Yue En

Aliatulmazwin Binti Maizul Azry

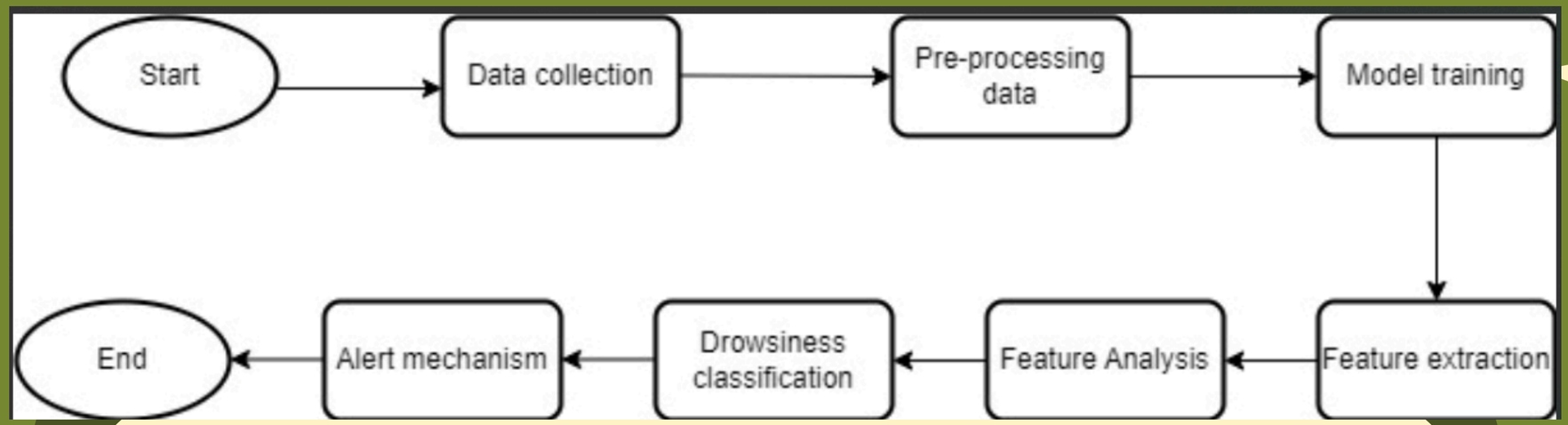
Link YouTube

https://youtu.be/sOzTbBWh9Bk?si=yQOWp42tO3iMgXwg

PROBLEM BACKGROUND

Drowsiness detection is critical for avoiding accidents and keeping people safe. This project uses image processing to detect indicators of tiredness, such as slow eye movements or head nodding, by analysing video captured with a camera. Drowsiness is difficult to quantify and varies by individual. Improving this technology can help us recognise when someone is overly drowsy, which is especially important for drivers and workers, and make the world safer.

METHODOLOGY



The methodology for drowsiness detection involves collecting and annotating video data to identify signs of drowsiness, such as slow eye movements and head nodding. Frames are extracted, and face and facial landmarks are detected to calculate features like the Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR). These features help analyze blink rate, yawning, and head movements. Machine learning or deep learning models are trained on these features to classify drowsiness.

PROGRAM CODES

```
% Initialize the video input from the second webcam
cam = webcam('Poly Studio P5 webcam');

% Create face, eye, and mouth detectors
faceDetector = vision.CascadeObjectDetector;
eyeDetector = vision.CascadeObjectDetector('EyePairBig', 'MergeThreshold', 20);
mouthDetector = vision.CascadeObjectDetector('Mouth', 'MergeThreshold', 10); % Lower threshold for better detection

% Initialize the video player to display the results
videoPlayer = vision.VideoPlayer('Position', [100, 100, 640, 480]);

% Define thresholds
drowsinessThresholdEyeFaceRatio = 0.085; % Threshold for eye area to face area ratio
drowsinessThresholdMouthFaceRatio = 0.09; % Threshold for mouth area to face area ratio
consecutiveDrowsyFrames = 20; % Number of consecutive frames to confirm drowsiness
headNodThreshold = 10;        % Threshold for vertical movement to detect nodding
nodCountThreshold = 3;        % Number of nods to confirm drowsiness
eyeDrowsyThreshold = 5;
mouthDrowsyThreshold = 5;
```

```
% Initialize counters and tracking variables
%drowsyFramesCounter = 0;
previousFaceBox = [];
verticalMovements = [];
nodCount = 0;
eyeDrowsyCount = 0;
mouthDrowsyCount = 0;
totalDrowsinessEvents = 0;

% Run the detection in a loop
while true
    % Capture a frame from the webcam
    img = snapshot(cam);

    % Convert to grayscale
    grayImg = rgb2gray(img);

    % Detect faces
    bbox = step(faceDetector, grayImg);
```

PROGRAM CODES

```
% If faces are detected, proceed
if ~isempty(bbox)
    % Choose the largest face detected
    [~, idx] = max(bbox(:, 3)); % Choose the face with the largest width
    faceBox = bbox(idx, :);

    % Calculate face area
    faceArea = faceBox(3) * faceBox(4);

     % Annotate the mouth with the mouth area value
     faceLabelPos = [faceBox(1), faceBox(2) - 20]; % Position above the mouth box
     img = insertText(img, faceLabelPos, sprintf('Face Area: %d', faceArea), 'FontSize', 12, 'BoxColor', 'green', 'BoxOpacity', 0.7, 'TextColor', 'white');

    % Check for nodding by comparing vertical positions of the face
    if ~isempty(previousFaceBox)
        verticalMovement = abs(faceBox(2) - previousFaceBox(2));
        verticalMovements = [verticalMovements, verticalMovement];

        % Check if the vertical movement exceeds the threshold
        if verticalMovement > headNodThreshold
            nodCount = nodCount + 1;
        else
            nodCount = max(nodCount - 1, 0); % Decay the nod count slowly
        end
```

```
    % Keep track of the last 15 vertical movements
    if length(verticalMovements) > 15
        verticalMovements = verticalMovements(2:end);
    end

    % If enough nods detected in recent frames, classify as drowsy
    if sum(verticalMovements > headNodThreshold) > nodCountThreshold
        %drowsyFramesCounter = drowsyFramesCounter + 1;
        label = 'Possible Drowsiness detected';
        color = 'yellow';
        load chirp.mat % load sound
        sound(y) % produce sound
        totalDrowsinessEvents = totalDrowsinessEvents +1;

    %else
        %drowsyFramesCounter = 0;
        %totalDrowsinessEvents = 0;
    end
end
previousFaceBox = faceBox;
```

PROGRAM CODES

```
% Extract the region of interest (ROI) containing the face
faceROI = imcrop(grayImg, faceBox);

% Initialize drowsiness detection label and color
label = 'Alert and awake!';
color = 'green';

% Flag to indicate drowsiness
isDrowsy = false;

% Detect eyes within the face ROI
eyeBBox = step(eyeDetector, faceROI);

% If eyes are not detected, assume they are closed
if isempty(eyeBBox)
    label = 'Eyes not detected';
end
```

```
% Ensure only one eye box is selected
if ~isempty(eyeBBox)
    % Select the box with the largest area if multiple eye boxes are detected
    if size(eyeBBox, 1) > 1
        [~, maxIdx] = max(eyeBBox(:, 3) .* eyeBBox(:, 4));
        eyeBBox = eyeBBox(maxIdx, :);
    end

    % Calculate eye area
    eyeArea = eyeBBox(3) * eyeBBox(4);

    % Adjust the eye bounding box position relative to the original image
    adjustedEyeBBox = adjustBBox(eyeBBox, faceBox);

    % Annotate the eyes on the image
    img = insertShape(img, 'Rectangle', adjustedEyeBBox, 'Color', 'blue');

    % Annotate the eyes with the eye area value
    eyeLabelPos = [adjustedEyeBBox(1), adjustedEyeBBox(2) - 20]; % Position above the eyes box
    img = insertText(img, eyeLabelPos, sprintf('Eye Area: %d', eyeArea), 'FontSize', 12, 'BoxColor', 'blue', 'BoxOpacity', 0.7, 'TextColor', 'white');
```

PROGRAM CODES

```
    % Calculate eye area to face area ratio
    eyeFaceRatio = eyeArea / faceArea;

    % Check if eye area to face area ratio is below the threshold
    if eyeFaceRatio < drowsinessThresholdEyeFaceRatio
        %isDrowsy = true;
        eyeDrowsyCount = eyeDrowsyCount + 1;
    else
        eyeDrowsyCount = max(eyeDrowsyCount - 1, 0);
    end

     % If enough eye drowsy detected in recent frames, classify as possible drowsy
    if eyeDrowsyCount > eyeDrowsyThreshold
        %drowsyFramesCounter = drowsyFramesCounter + 1;
        label = 'Possible Drowsiness detected';
        color = 'yellow';
        load chirp.mat % load sound
        sound(y) % produce sound
        totalDrowsinessEvents = totalDrowsinessEvents +1;
    end
end
```

```
% Limit the search area for the mouth to the lower half of the face ROI
mouthROI = imcrop(faceROI, [1, faceBox(4)/2, faceBox(3), faceBox(4)/2]);
mouthBBox = step(mouthDetector, mouthROI);

if isempty(mouthBBox)
    label = 'Mouth not detected';
end

% If mouth is detected, calculate mouth area and adjust bounding box
if ~isempty(mouthBBox)
    % Ensure only one mouth box is detected
    if size(mouthBBox, 1) > 1
        % Select the box with the largest area
        [~, maxIdx] = max(mouthBBox(:, 3) .* mouthBBox(:, 4));
        mouthBBox = mouthBBox(maxIdx, :);
    end

    % Adjust mouth bounding box coordinates to the original face ROI
    mouthBBox(1:2) = mouthBBox(1:2) + [faceBox(1), faceBox(2) + faceBox(4)/2];

    % Calculate mouth area
    mouthArea = mouthBBox(3) * mouthBBox(4);
```

PROGRAM CODES

```
% Annotate the mouth on the image
img = insertShape(img, 'Rectangle', mouthBBox, 'Color', 'cyan');

% Annotate the mouth with the mouth area value
mouthLabelPos = [mouthBBox(1), mouthBBox(2) - 20]; % Position above the mouth box
img = insertText(img, mouthLabelPos, sprintf('Mouth Area: %d', mouthArea), 'FontSize', 12, 'BoxColor', 'cyan', 'BoxOpacity', 0.7, 'TextColor', 'white');

% Calculate mouth area to face area ratio
mouthFaceRatio = mouthArea / faceArea;

% Check if mouth area to face area ratio is below the threshold
if mouthFaceRatio > drowsinessThresholdMouthFaceRatio
    % isDrowsy = true;
    mouthDrowsyCount = mouthDrowsyCount + 1;
else
    mouthDrowsyCount = max(mouthDrowsyCount - 1, 0);
end
```

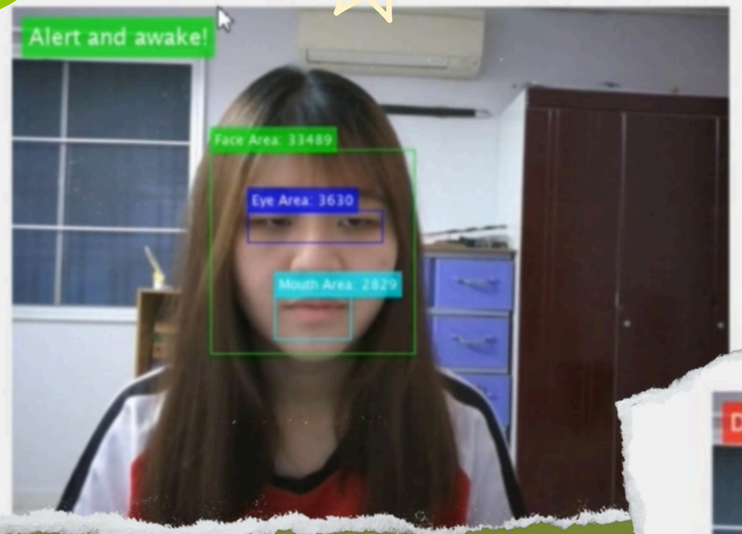
```
    % If enough eye drowsy detected in recent frames, classify as possible drowsy
    if mouthDrowsyCount > mouthDrowsyThreshold
        %drowsyFramesCounter = drowsyFramesCounter + 1;
        label = 'Possible Drowsiness detected';
        color = 'yellow';
        load chirp.mat % load sound
        sound(y) % produce sound
        totalDrowsinessEvents = totalDrowsinessEvents +1;
    end
end

% % Update drowsiness counter if any drowsiness condition is met
% if isDrowsy || drowsyFramesCounter > 0
%     drowsyFramesCounter = drowsyFramesCounter + 1;
% else
%     drowsyFramesCounter = 0;
% end
```

PROGRAM CODES

```
    % Set label and color if drowsiness detected over consecutive frames
    if totalDrowsinessEvents >= consecutiveDrowsyFrames
        label = 'Drowsiness detected!';
        color = 'red';
        load gong.mat % load sound
        sound(y) % produce sound
        totalDrowsinessEvents = 0; % Reset the drowsiness events counter
    end

    % Annotate the face and the detection result
    img = insertText(img, [10, 10], label, 'FontSize', 20, 'BoxColor', color, 'BoxOpacity', 0.7, 'TextColor', 'white');
    img = insertShape(img, 'Rectangle', faceBox, 'Color', color);
else
    % If no face is detected, display a message
    img = insertText(img, [10, 10], 'Face not detected!', 'FontSize', 20, 'BoxColor', 'yellow', 'BoxOpacity', 0.7, 'TextColor', 'black');
    verticalMovements = [];
    nodCount = 0;
    drowsyFramesCounter = 0;
    totalDrowsinessEvents = 0; % Reset the drowsiness events counter
    eyeDrowsyCount = 0;
    mouthDrowsyCount = 0;
end
```

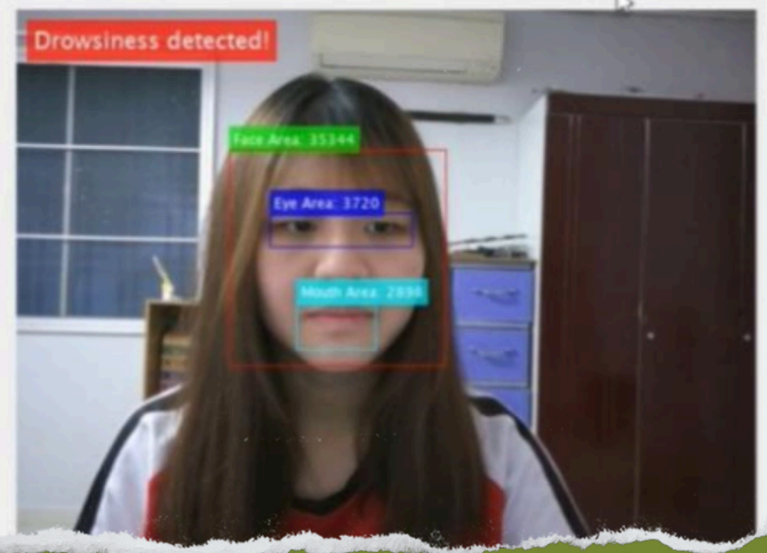
```
    % Display the annotated video frame
    step(videoPlayer, img);

    % Exit the loop if the video player window is closed
    if ~isOpen(videoPlayer)
        break;
    end
end
% Release resources
release(videoPlayer);
clear cam;

% Function to adjust bounding box coordinates relative to original image
function adjustedBBox = adjustBBox(bbox, offset)
    % Adjust the bounding box coordinates to account for the offset
    adjustedBBox = bbox;
    adjustedBBox(1) = bbox(1) + offset(1);
    adjustedBBox(2) = bbox(2) + offset(2);
end
```

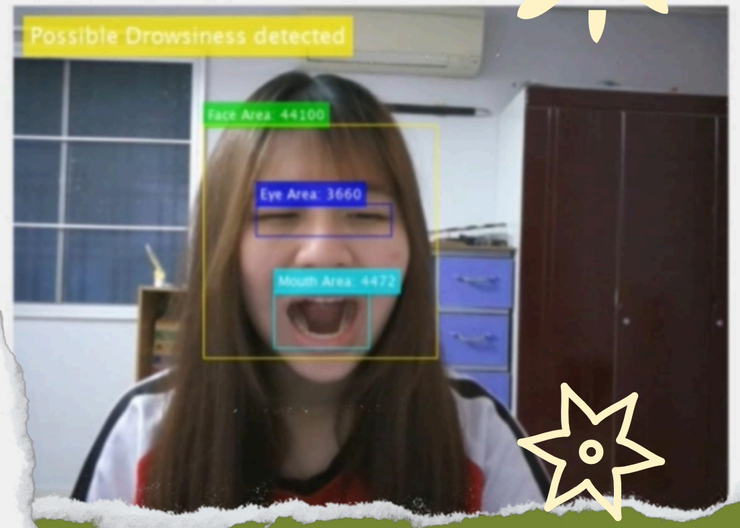
OUTPUT



ALERT AND AWAKE

DROWSINESS DETECTED





POSSIBLE DROWSINESS DETECTED

CONCLUSION

Drowsiness detection utilising image processing helps to avoid accidents by detecting early symptoms of tiredness. The technology examines video from a camera to recognise features such as eyes and mouth, looking for indicators like closed eyes or yawning. It also detects head nodding. If drowsiness is identified, it notifies the user in real time. This strategy is critical for improving safety, particularly for drivers and workers, as it reduces the likelihood of fatigue-related accidents.

REFERENCES

1. Drowsiness detection using a binary SVM classifier. File Exchange - MATLAB Central. (n.d.). https://www.mathworks.com/matlabcentral/fileexchange/55152-drowsiness-detection-using-a-binary-svm-classifier

2. Saini, V., & Saini, R. (2014). Driver drowsiness detection system and techniques: a review. International Journal of Computer Science and Information Technologies, 5(3), 4245-4249.

3. Drowsiness detection system. (2022, May). Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/05/drowsiness-detection-system/

4. Yang, Y., Liu, X., He, Z., Li, J., Yang, J., Li, X., ... & Zhang, L. (2022). A drowsiness detection system based on facial expression recognition using a deep neural network. PLOS ONE, 17(3), e0265211. https://doi.org/10.1371/journal.pone.0265211

5. Driver drowsiness detection. (n.d.). Bosch Mobility Solutions. https://www.bosch-mobility.com/en/solutions/assistance-systems/driver-drowsiness-detection/

THANK YOU