CV Mini Project

Name: Abhay Mathur
SAPID: 60017210016
Branch: AIML A1

Theory:

In this project, we have implemented a real-time hand gesture-controlled ping-pong game using the cvzone library for hand tracking and OpenCV for image processing. First, we capture video from our computer's camera and detect hands using a hand detector. Then, we resize the game window for better viewing. We import images for the game elements like the background, ball, bats, and game over screen. Within the game logic, we handle bat movement based on hand position, ball movement, collision detection, scoring, and game over conditions. The game loop continuously updates the frame with the game elements and displays the score. If we want to restart the game, we just need to press 'R', and to quit, we press 'Q'.

Code:

pong.py:

```
import numpy as np
import cv2 as cv
import cvzone
from cvzone.HandTrackingModule import HandDetector

# Captures the video feed from your computer's camera. 0 denotes the camera of
your laptop.
capture = cv.VideoCapture(0)
# Used to detect maximum two hands with confidence 0.8.
detector = HandDetector(detectionCon = 0.8, maxHands = 2)

# Usually, the game window pop-up will have 640 x 480 resolution. This is far too
small an area for our game.
# So, we have to resize our window.
# 3 denotes your width. It sets the width of the game window to 1280 pixels.
capture.set(3, 1280)
# 4 denotes your height. It sets the height of the game window to 720 pixels.
capture.set(4, 720)

# Importing all images. imread() helps to read an image from a path.
# cv.IMREAD_UNCHANGED is to ensure the image is read without any preprocessing
being done by the imread() function.
# This is needed for the transparency of the images to be maintained for the
"tools" of the game, so to say.
background = cv.imread(r'C:\Users\a21ma\OneDrive\Desktop\Code\Machine
Learning\Ping Pong Game\Resources\Background.png')
# background = cv.resize(background, 1280, 720)
ball = cv.imread(r'C:\Users\a21ma\OneDrive\Desktop\Code\Machine Learning\Ping Pong
Game\Resources\Ball.png',cv.IMREAD_UNCHANGED)
bat1 = cv.imread(r"C:\Users\a21ma\OneDrive\Desktop\Code\Machine Learning\Ping Pong
Game\Resources\bat1.png",cv.IMREAD_UNCHANGED)
bat2 = cv.imread(r"C:\Users\a21ma\OneDrive\Desktop\Code\Machine Learning\Ping Pong
Game\Resources\bat2.png",cv.IMREAD_UNCHANGED)
```

```
gameOver = cv.imread(r'C:\Users\a21ma\OneDrive\Desktop\Code\Machine Learning\Ping Pong Game\Resources\gameOver.png')

# print(background.shape)
# print(ball.shape)
# print(bat1.shape)
# print(bat2.shape)
# print(gameOver.shape)
# Set initial ball postion, score, a flag and the speeds
position = [100, 100]
isOver = False
speedX, speedY = 15, 15
score = [0, 0]

# To capture video footage from the camera.
while True:
    # read() function returns the frame and the boolean value that says whether the frame was read successfully or not.
    # We flip the image horizontally to avoid confusion due to lateral inversion.
    isTrue, frame = capture.read()
    frame = cv.flip(frame, 1)
    frame = cv.resize(frame,(1280,720))
    # To detect hands and its landmarks, without any annotations. findHands returns a boolean value.
    hands, frame = detector.findHands(frame, flipType = False)

    # For overlaying images.
    # addWeighted() puts the foreground image with an amount of transparency on top of the underlying image.
    # In this case, our underlying image at every instant is the camera feed.
    frame = cv.addWeighted(frame, 0.5, background, 0.5, 0.0)

    # Check for hands. Hands is a dictionary, so we process it like one.
    if hands:
        for hand in hands:
            # Gets the positional values of hand in bounding box.
            x, y, w, h = hand['bbox']
            h1, w1, _ = bat1.shape  # To get height and width of the bat
            y1 = y - h1//2  # To ensure your hands are at center of the bat.
            # Clips the value to ensure overlays don't occur outside the pop-up space.
            y1 = np.clip(y1, 20, 415)

            if hand['type'] == 'Left':
                left_bat_pos = 59
                frame = cvzone.overlayPNG(frame, bat1, (left_bat_pos, y1))

                if (left_bat_pos< position[0] < left_bat_pos + w1) and (y1 < position[1] < y1 + h1):
                    # If bat hits the ball, reverse the ball direction along x-axis.
                    speedX = -speedX
                    position[0] += 20
```

```
                    score[0] += 1

            if hand['type'] == 'Right':
                frame = cvzone.overlayPNG(frame, bat2, (1195, y1))
                if (1145 < position[0] < 1165) and (y1 < position[1] < y1 + h1):
                    speedX = -speedX
                    position[0] -= 20
                    score[1] += 1

    # If ball goes out of bounds, game's over.
    if position[0] < 40 or position[0] > 1195:
        isOver = True

    if isOver:
        frame = gameOver
        cv.putText(frame, str(score[1] + score[0]).zfill(2), (585, 360),
cv.FONT_HERSHEY_COMPLEX,
                   2.5, (200, 0, 200), 5)
    else:
        # Move the ball. If the ball hits the wall of upper rectangle, reverse its
direction along y-axis.
        if position[1] >= 500 or position[1] <= 10:
            speedY = -speedY

        # We give the ball speed.
        position[0] += speedX
        position[1] += speedY

        # print(frame.shape)
        # Draw the ball
        frame = cvzone.overlayPNG(frame, ball, position)
        # Display score as game goes on. we use putText() for this. (255, 255,
255) is white colour
        cv.putText(frame, str(score[0]), (300, 650),
                   cv.FONT_HERSHEY_COMPLEX, 3, (255, 255, 255), 5)
        cv.putText(frame, str(score[1]), (900, 650),
                   cv.FONT_HERSHEY_COMPLEX, 3, (255, 255, 255), 5)

    # imshow() function returns the matrix of pixels in a new window. It also
takes the name of the popup window.
    cv.imshow('Pong game', frame)
    # waitKey() function allows the popup window to be displayed for a certain
time period in milliseconds.
    key = cv.waitKey(1)
    # If R is pressed, re-initialise the respective variables and restart the
game.
    if key == ord('r'):
        position = [100, 100]
        speedX = 15
        speedY = 15
        isOver = False
        score = [0, 0]
```

```
        gameOver = cv.imread(r'C:\Users\a21ma\OneDrive\Desktop\Code\Machine
Learning\Ping Pong Game\Resources\gameOver.png')

    # To quit the game
    if key == ord('q'):
        break

# Release the capture and close all OpenCV windows
capture.release()
cv.destroyAllWindows()
```

Output: