**CS585 Assignment 2**

**Video Processing & Shape Analysis**

**A hand in a hexagon shape

Description automatically generated**

Report By: Munir Siddiqui

Teammate: Muhammad Aseef Imran

14 February 2024

1. **Problem Definition**
2. **Description:**

The problem at hand is to design and implement an algorithm that recognizes hand shapes or gestures, and to create a graphical display that responds to the recognition of the hand shapes or gestures. This is a task with broad applications in human-computer interaction, accessibility, gesture-based control systems, entertainment, and healthcare. By enabling users to interact with devices and systems using natural hand movements, gesture recognition enhances user experience, accessibility for individuals with disabilities, and usability in various environments where traditional input methods are impractical.

1. **Our Implementation:**

The primary challenge is to accurately detect and interpret hand gestures captured by a webcam in real-time. We decided to implement an algorithm that detects the number of fingers raised in one hand by a person, and then use that algorithm to play a fun memory game. Our goal is to recognize hand gestures captured by a webcam, interpret them as numbers, and then compare the guessed number with a randomly generated number for each round of the game. This involves several key aspects:

1. Hand Detection: The algorithm needs to identify and isolate the hand region from the background and other objects in the webcam feed. Accurate hand detection is essential for subsequent analysis and recognition of gestures.
2. Finger Detection: The algorithm must accurately count the number of fingers visible in the detected hand region. Finger detection involves analyzing the contours and shapes within the hand region to infer the positions and orientations of individual fingers.
3. Gesture Interpretation: Once the fingers are detected, the system must interpret their configuration as numeric values. Each numeric value corresponds to a specific hand gesture, representing numbers from 1 to 5. The accuracy of gesture interpretation is crucial for the player to correctly guess the randomly generated number in the memory game.
4. Real-time Performance: The system must operate in real-time, processing webcam frames at a sufficient speed to provide a smooth and interactive user experience. Achieving real-time performance requires efficient algorithms and optimized code implementation.
5. **Assumptions:**

Our algorithm makes the following assumptions, all of which are related to the hand-detection part of the algorithm:

1. The background does not contain any skin color, such as wood or skin-colored walls.
2. The lighting condition is ideal, so that the hand is illuminated properly and can be detected as having skin color.
3. The user’s face is not in the frame. If it is, then it must not overlap with the hand and in perspective, it must be smaller than the user’s hand. In other words, the user’s hand must be the largest isolated skin-colored object in the frame.
4. The user is wearing full sleeves, as the wrist causes problems with the hand-detection part of the algorithm.
5. The user’s hand is rotated between 0° and 90° clockwise/anticlockwise.
6. **Anticipated Difficulties:**

We anticipated that the following tasks would be challenging:

1. Determining where the hand ends and the wrist begins.
2. Finding out the orientation of the hand and rotating it back to 0°.
3. Identifying the number of fingers raised in the hand.
4. **Method and Implementation**

The implemented method involves several steps. The helper function which implements each step is written in parentheses after each step:

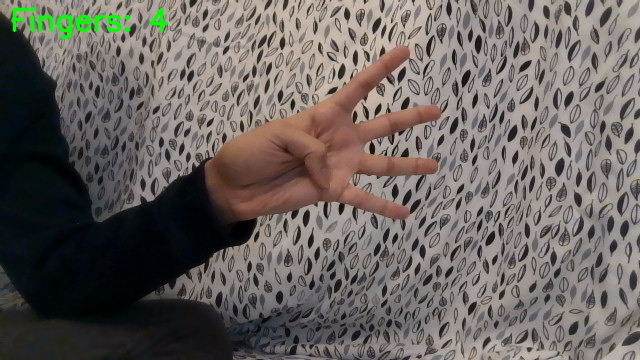
1. **count\_fingers():**
   1. **Preprocessing:**

* Blurring to remove noise. (apply\_smoothing)
* Masking to isolate skin-colored objects. (mask\_image)
* Conversion to a binary image. (convert\_to\_binary)
* Dilation to smoothen the edges of the binary image. (no separate helper function)
  1. **Object Extraction:**
* Using contouring, finding the largest object in the binary image, assumed to be the hand. (binary\_img\_extract\_largest\_obj)
* Refining the object by reducing contour complexity and removing defects. (angle\_contour\_reducer, defects\_remover\_via\_angle\_checking)
* Dilation once again to reduce further defects. (no separate helper function)
* Finding largest object in the binary image once again – this helps to fill in the holes inside the binary image of the hand. (binary\_img\_extract\_largest\_obj)
  1. **Analysis:**
* Scaling the object so that the hand does not get cropped when it is later rotated. (scale\_obj)
* Finding the axis of least inertia to determine the orientation of the hand. (find\_axis\_of\_least\_inertia)
* Translating the hand to the center of the image to facilitate rotation. (move\_obj\_to\_center)
* Rotating the hand to 0° based on the calculated orientation. (rotate\_at\_center)
  1. **Recognition:**
* Finding the largest object yet again in the scaled and rotated image. (get\_fingers\_in\_frame)
* Finding the bounding box that completely bounds the hand. (get\_fingers\_in\_frame)
* Drawing a circle of radius equal to one third of the height of the bounding box (determined experimentally), centered at the centroid of the hand. (get\_fingers\_in\_frame)
* Counting the number of fingers intersecting with the circle by considering transitions from background to foreground pixels and vice versa. (get\_fingers\_in\_frame)
* Calculating the circularity of the hand to distinguish between four and five fingers. (calc\_roundness)
* Using the mean and standard deviation of the number of fingers counted to eliminate anomalies. (no separate helper function)

1. **main():**
   1. **Game Logic:**

* Displaying the graphics for the game
* Generating a random number for the player to memorize.
* Capturing the player's guessed number based on hand gestures. (count\_fingers)
* Comparing the guessed number with the generated number to determine correctness.

Here are some images of how the hand looks during each step of the algorithm:

A hand with fingers spread out

Description automatically generated

Fig -Original frame; user sees this during the game

Fig - Image is blurred to reduce noise

A white hand print on a black background

Description automatically generatedA hand with a hole in the middle

Description automatically generated

Fig 4 - Colored image is converted to a binary image

Fig 3 - Masked image with only skin-color pixels remaining

A white hand on a black background

Description automatically generatedA white hand print on a black background

Description automatically generated

Fig 6 - Largest object extracted from the image

Fig 5 - Image after first dilation

A white hand on a black background

Description automatically generatedA white hand on a black background

Description automatically generated

Fig 8 - Image after second dilation

Fig 7 - Image after reducing contours and removing defects

A hand with a green line

Description automatically generatedA white hand on a black background

Description automatically generated

Fig 10 - Scaled image with axis of least inertia marked

Fig 9 - Largest object extracted from the image again

A handprint with green circle

Description automatically generatedA white hand print on a black background

Description automatically generated

Fig 12 – Final image with circle drawn to count the number of fingers

Fig 11 – Image is centered and rotated to 0°

1. **Experiments**
2. **Description:**

The problem at hand is to design and implement an algorithm that recognizes hand shapes or gestures, and to create a graphical display that responds to the recognition of the hand shapes or gestures. This is a task with broad applications in human-computer interaction, accessibility, gesture-based control systems, entertainment, and healthcare. By enabling users to interact with devices and systems using natural hand movements, gesture recognition enhances user experience, accessibility for individuals with disabilities, and usability in various environments where traditional input methods are impractical.

1. **Our Implementation:**

The primary challenge is to accurately detect and interpret hand gestures captured by a webcam in real-time. We decided to implement