CAS CS 585 - Spring 2024

Assignment 2

Teammates:
1. Uday Garg- U11302797
Date Of Submission: 02/14/2024

Problem Definition:

To design and build an algorithm that can recognize hand gestures or shapes and create a graphical display that responds to the recognition of the hand shapes or gestures.

Method and Implementation:

My code implements a hand gesture recognition method using computer vision with OpenCV. The primary goal of this method is to allow a computer to interpret specific human hand gestures by analyzing video input in real-time.

Algorithmic Steps:

1. Skin Detection: The script starts by capturing a video frame and converting it from BGR to HSV color space. This color space conversion is crucial for effectively isolating skin tones from the rest of the image. A predefined range of HSV values, known to represent skin colors, is used to create a binary mask that highlights regions of the frame corresponding to human skin.

2. Hand Contour Detection: With the skin mask obtained, the script then searches for contours within this mask. Contours are essentially boundaries of connected regions with the same color or intensity. Among all detected contours, the one with the largest area is assumed to represent the hand. This assumption is based on the expectation that the hand is the primary subject of interest and thus the largest skin-toned object in the frame.

3. Feature Extraction: Upon identifying the hand contour, the script proceeds to analyze its shape to extract meaningful features. This involves computing the convex hull of the contour, which is the smallest convex shape that encompasses all points of the contour. Convexity defects, which are deviations of the contour from the convex hull, help identify the valleys between extended fingers. By analyzing these defects and the angles formed at their points, the script determines the number of fingers extended and identifies the presence of a thumb.

4. Gesture Recognition: Based on the extracted features, the script classifies the observed hand posture into predefined gestures. It uses simple rules, such as the count of extended fingers and the detection of a thumb, to categorize the gesture as "Open Hand", "Pointing", "Peace", "Fist", or "Unknown" if it doesn't match any predefined category.

Implemented Functions:

- `detect_skin(frame)`: Converts the input frame to HSV color space and applies a mask to isolate skin-colored regions, aiming to highlight the hand.
- `find_hand_contours(mask)`: Identifies contours in the skin mask and selects the largest one, which is presumed to be the hand contour.

- `extract_features(contour)`: Analyzes the detected hand contour, computing the convex hull and identifying convexity defects to extract features such as the number of extended fingers and the presence of a thumb.
- `recognize_gesture(features)`: Maps the extracted features to predefined gestures, returning a descriptive string of the recognized gesture.
- `main()`: Orchestrates the overall process, capturing video frames, processing them to detect gestures, and displaying the results with visual cues on the frame. It also provides a mechanism for the user to exit the program.

This method effectively combines several computer vision techniques to interpret hand gestures, relying on skin color detection, contour analysis, and geometric features extraction to recognize predefined hand gestures in real-time video input.

Experiments:

Throughout the development process I aimed at devising a robust algorithm for hand gesture recognition, a series of methodological explorations and iterative refinements were undertaken. The objective was to establish a consistent and accurate system capable of interpreting various hand gestures. Below is a detailed account of the experimental phases and the rationale behind the progression of methodologies:

Initial Approach: Circularity and Aspect Ratio Analysis:

The preliminary strategy involved preprocessing the input image by converting it to grayscale and applying a Gaussian blur to diminish noise, followed by skin detection and feature extraction phases. A key metric employed was the circularity of detected contours, utilized to distinguish between an open hand and a fist. While this naive approach yielded high success rates, it was inherently constrained to recognizing only two gestures.

Template Matching Exploration:

In an effort to expand the gesture count, template matching was implemented as a potential technique. Despite the use of various template sources, including standard stock images and high-resolution photographs of my own hands, this method failed to produce satisfactory results. The inherent variability in hand positioning, orientation, and individual differences rendered template matching ineffective, leading me to abandon it.

Convexity Analysis and Defects-Based Method:

The third phase of experimentation saw the adoption of convexity analysis, specifically through the examination of convexity defects within hand contours. This technique facilitated the identification of finger gaps, enabling the recognition of finger-based gestures such as the peace sign, in addition to the previously recognized fist and open hand gestures. This advancement marked a significant improvement, but still limited my flexibility to include more gestures.

Integration of Thumb Detection:

To further enrich the gesture recognition capabilities, thumb detection was incorporated into the feature extraction process. Initially, thumb detection alone proved insufficient for reliable gesture classification. However, when combined with finger count analysis, a more comprehensive and nuanced understanding of hand postures was achieved. This integrated approach culminated in the recognition of four distinct gestures, like open hand, fist, peace sign, and pointing.

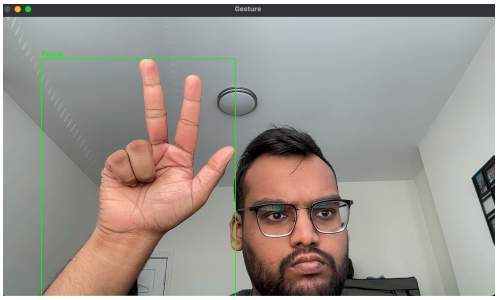
Results:

Accuracy = **0.74**

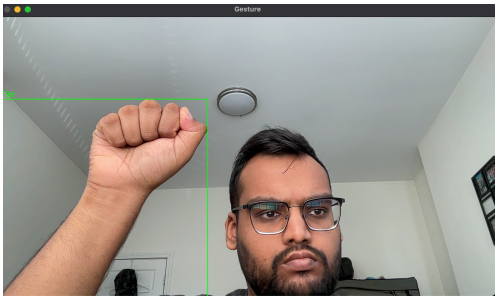
F1 = **0.741**

Recall = **0.740**

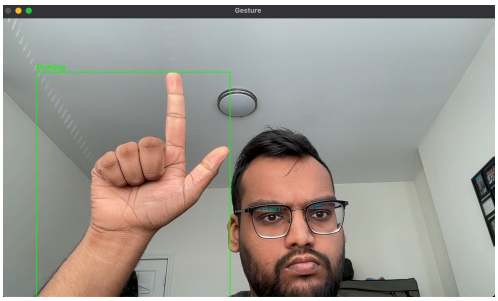
Precision = **0.803**



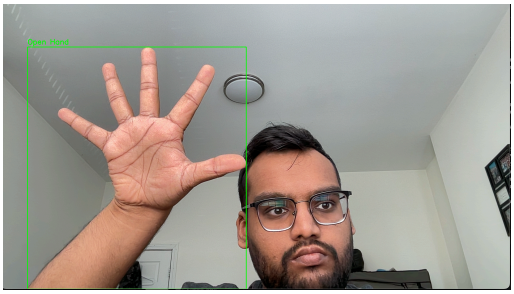
Peace Sign



Fist



Pointing



Open Hand

PREDICTED V GROUND TRUTH	FIST	OPEN HAND	POINTING	PEACE	UNKNOWN
FIST	8	0	0	0	2
OPEN HAND	0	6	0	1	3
POINTING	1	0	6	2	1
PEACE	2	0	0	7	1
UNKNOWN	0	0	0	0	10

Confusion Matrix with Convexity defects and thumb detection.

Discussion:

- Strengths
  - Gesture Recognition Based on Convexity Defects:

The method's reliance on convexity defects for feature extraction enables the recognition of distinct gestures by analyzing the shape and structure of the hand contour. This approach is less susceptible to variations in background and lighting compared to template matching.

- Weaknesses and Limitations
  - Limited Gesture Set:

The current implementation is capable of recognizing only four gestures, constrained by the method's reliance on finger count and thumb detection. This limits the system's applicability in more complex communication systems.

  - Sensitivity to Hand Orientation, Position and Lighting:

The accuracy of gesture recognition can be significantly impacted by the hand's orientation and position relative to the camera. Non-ideal angles or partial occlusions can lead to misinterpretation of gestures. Sensitivity to light causes the algorithm to “loose” the hand and misreport the gesture.

- Potential Future Work
  - User-Centered Adaptation:

Implementing mechanisms for user-specific calibration and adaptation could address variations in hand size, skin tone, and individual gesture nuances, making the system more personalized and user-friendly.

  - Deep Learning Approaches:

Incorporating deep learning, particularly convolutional neural networks (CNNs), could significantly enhance the system's ability to recognize a more extensive set of gestures. CNNs are capable of learning complex patterns and variations in hand postures, providing a more generalized and robust approach to gesture recognition.

  - Improved Hand Tracking and Orientation Analysis:

Developing advanced algorithms for hand tracking and orientation analysis could mitigate issues related to hand positioning and partial occlusions, thereby improving recognition accuracy.

Conclusion:

The iterative exploration and refinement of hand gesture recognition techniques underscored the complexity and challenges inherent in achieving a versatile and reliable recognition system. Through a process of trial, error, and continuous adaptation, a method combining convexity defect analysis with thumb detection emerged as a promising solution, offering a balance between accuracy and the diversity of recognizable gestures. This journey highlights the importance of adaptability and the willingness to explore a spectrum of approaches in the quest for optimal solutions in the field of computer vision and gesture recognition.

Credits and Bibliography:

1. Class notes: https://www.cs.bu.edu/faculty/betke/cs585/open/2024-cs585-projections-localization-floodfill-Jan25.pdf
2. OpenCV documentation: https://docs.opencv.org/4.x/index.html
3. Convexity defects: https://docs.opencv.org/4.x/d8/d1c/tutorial_js_contours_more_functions.html
4. ChatGPT - Report proofreading.
    1. SAMPLE PROMPT: “Check the presented document for any grammatical errors”
    2. SAMPLE RESPONSE:

        I've reviewed the document for grammatical errors. Here's a summary of the key points that need attention:

        1. **Consistency in Tense:** Ensure that the report maintains a consistent tense throughout, particularly when discussing the methods, results, and conclusions. Switching between past and present tense can confuse the reader.

        2. **Subject-Verb Agreement:** Check for subject-verb agreement errors, especially in complex sentences with multiple clauses.

        3. **Punctuation:** Pay attention to the use of commas and periods, especially in long sentences to improve readability. Also, check the use of apostrophes for possessive forms.

        4. **Spelling and Typographical Errors:** Look for any misspelled words or typographical errors.

        5. **Sentence Structure:** Ensure sentences are not too long or complex, which can make the report hard to follow. Breaking down complex ideas into simpler sentences can enhance clarity.

        6. **Use of Articles:** Review the use of definite and indefinite articles ('the', 'a', 'an') to ensure they are used correctly.

        7. **Consistency in Formatting:** Ensure consistency in formatting terms, code elements, and other specialized content throughout the document.

        8. **Clarity and Conciseness:** Aim for clarity and conciseness in your explanations. Avoid unnecessary jargon and ensure that each sentence contributes to your overall argument or explanation.