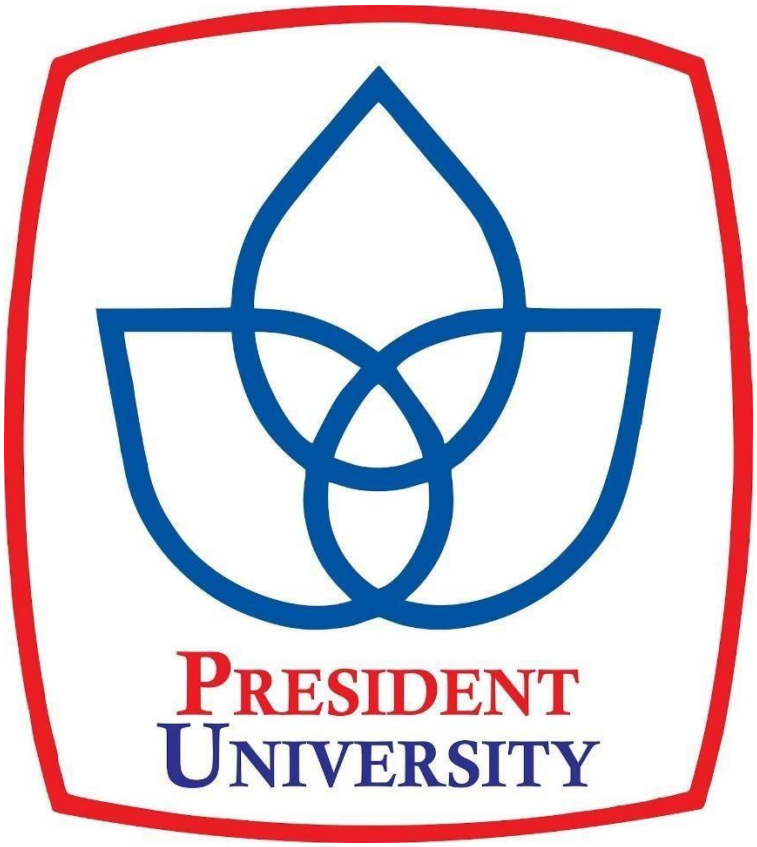
PROJECT COMPUTER VISION

“Single Pose Detection”



Fitri Anggraini

(001201900071)

1. Introduction.

- What is the program about?

  → This program presents a detection methodology to monitor human movement and activity using deep learning and computer vision as well as MoveNet which can detect 17 key points of the body. This pose estimation opens applications in various fields, such as Augmented reality, Animation, Games, and Robotics. The goal is to approximate some human poses in the given image. By using pre-recorded video or a webcam as input and an open-source object detection model using the MoveNet algorithm can detect human movements and activities. It also works on web cameras, CCTV, etc., and can detect people in real-time.

- In what language is the program implemented?

  → The program that we created uses the Python language to implement the program. Python is a high-level general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

  → Detect MoveNet is a real-time person detection library, and it's the library has shown the functionality of collectively detecting the accurate model that detects 17 keypoints of a body.

  → Process images and videos to identify objects, faces of a human using OpenCV. OpenCV is a great tool for image processing and performing computer vision tasks.

  → We are using the TensorFlow for implementing machine learning and deep learning applications.

2. Methodology.

- Input Collection.

  → Images and videos recorded by CCTV cameras are provided as input, or you can also use a webcam directly.

- Calibrating the camera.

  → The region of interest (ROI) of an image or a video frame focused on the person who is walking was captured using a CCTV camera or Webcam. The calibration is done by transforming the view frame captured into a two-

dimensional bird's view. The camera calibration is done straightforwardly using OpenCV. The transformation of view is done using a calibration function that selects 4 points in the input image/video frame and then mapping each point to the edges of the rectangular two-dimensional image frame.

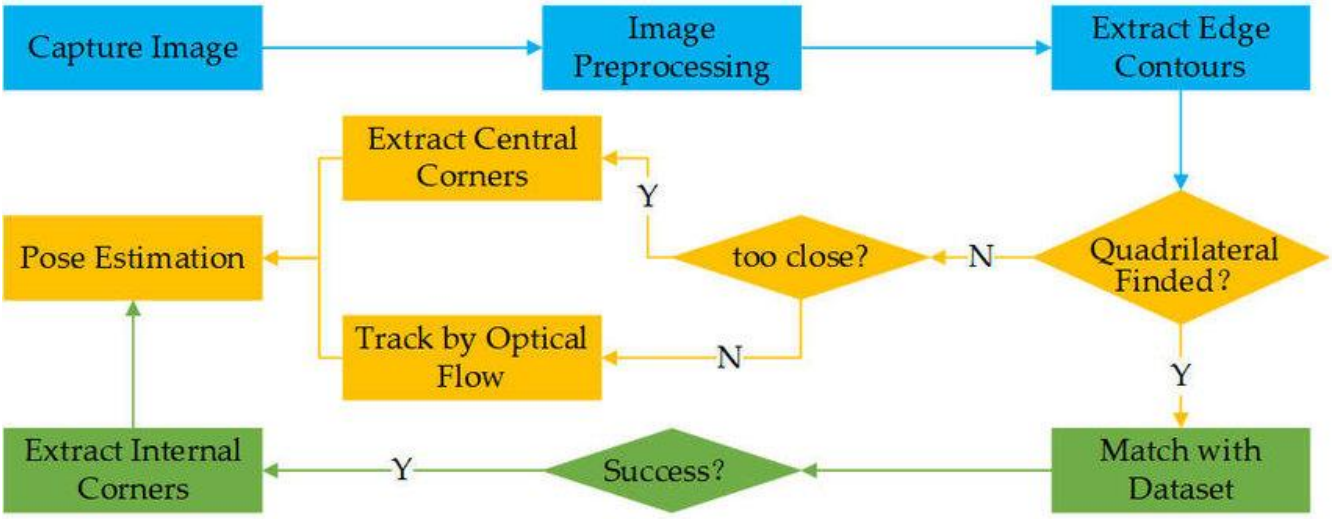
- Detection of Human

  → Deep Convolutional Neural Networks model is a simple and efficient model for object detection. This model considers the region which contains only "Person" class and discards the regions that are not likely to contain any object. The object detection approach used in the Pose Detection analyzer model reduces the computational complexity issues. It is done by formulating the detection of objects with the help of a single regression problem. In object detection models based on deep learning, Use the MoveNet and Tensorflow model. This model is suitable for real-time applications and it is faster and provides accurate results. The MoveNet and Tensorflow is an object detection model that takes an image or a video as an input and can simultaneously learn and draw Keypoint of body, corresponding class label probabilities, and object confidence.

- Output

  → After the input is processed and After MoveNet draws the keypoint of a body, then the output is displayed by OpenCv.

- Code Flow.



3. Explanation code SinglePoseDetection.py

- Importing Dependencies.

```
1  import tensorflow as tf
2  import tensorflow_hub as hub
3  import cv2
4  from matplotlib import pyplot as plt
5  import numpy as np
```

→ Here, we have to Import our important dependencies for the program.

- Loading the Model.

```
7  interpreter = tf.lite.Interpreter(model_path='lite-model_movenet_singlepose_lightning_3.tflite')
8  interpreter.allocate_tensors()
```

→ load model to detect and draw keypoint of body

- Drawing Keypoints and edges.

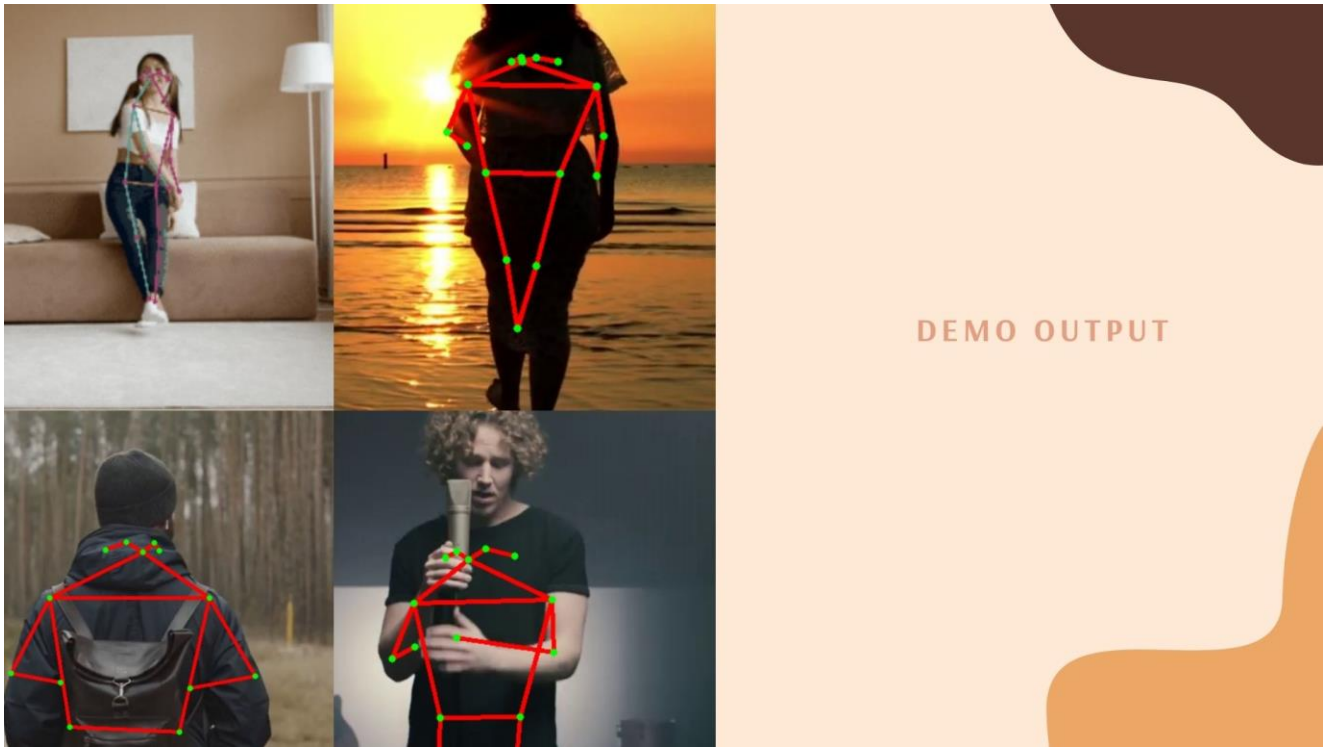
```
15 def draw_keypoints(frame, keypoints, confidence_threshold):
16     y, x, c = frame.shape
17     shaped = np.squeeze(np.multiply(keypoints, [y,x,1]))
18
19     for kp in shaped:
20         ky, kx, kp_conf = kp
21         if kp_conf > confidence_threshold:
22             cv2.circle(frame, (int(kx), int(ky)), 4, (0,255,0), -1)
23 def draw_connections(frame, keypoints, edges, confidence_threshold):
24     y, x, c = frame.shape
25     shaped = np.squeeze(np.multiply(keypoints, [y,x,1]))
26
27     for edge, color in edges.items():
28         p1, p2 = edge
29         y1, x1, c1 = shaped[p1]
30         y2, x2, c2 = shaped[p2]
31
32         if (c1 > confidence_threshold) & (c2 > confidence_threshold):
33             cv2.line(frame, (int(x1), int(y1)), (int(x2), int(y2)), (0,0,255), 4)
34
```

```python
EDGES = {
    (0, 1): 'm',
    (0, 2): 'c',
    (1, 3): 'm',
    (2, 4): 'c',
    (0, 5): 'm',
    (0, 6): 'c',
    (5, 7): 'm',
    (7, 9): 'm',
    (6, 8): 'c',
    (8, 10): 'c',
    (5, 6): 'y',
    (5, 11): 'm',
    (6, 12): 'c',
    (11, 12): 'y',
    (11, 13): 'm',
    (13, 15): 'm',
    (12, 14): 'c',
    (14, 16): 'c'
}
```

- Making Detection Using Video and Webcam

```python
cap = cv2.VideoCapture('Test.mp4')
while cap.isOpened():
    ret, frame = cap.read()

    # Resize image
    img = frame.copy()
    img = tf.image.resize_with_pad(tf.expand_dims(img, axis=0), 192,192)
    input_image = tf.cast(img, dtype=tf.float32)

    # Setup input and output
    input_details = interpreter.get_input_details()
    output_details = interpreter.get_output_details()

    # Make predictions
    interpreter.set_tensor(input_details[0]['index'], np.array(input_image))
    interpreter.invoke()
    keypoints_with_scores = interpreter.get_tensor(output_details[0]['index'])

    # Rendering
    draw_connections(frame, keypoints_with_scores, EDGES, 0.4)
    draw_keypoints(frame, keypoints_with_scores, 0.4)

    loop_through_people(frame,keypoints_with_scores, EDGES, 0.1)

    cv2.imshow('MoveNet Lightning', frame)

    if cv2.waitKey(10) & 0xFF==ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

4. Results



Here, you can see that I was able to process the entire and as the results show, our Single Pose detector is correctly marking people pose.

5. Conclusion.

A methodology of Single pose estimation detection tool using a deep learning model is proposed. By using computer vision, the motion and activities people can be estimated and any non-compliant pair of people will be indicated with a keypoint. The proposed method was validated using a video showing people on a street. The visualization results showed that the proposed method is capable of determining the Single pose estimation by drawing keypoint body of people.

References

G V Shalini et a | Journal of Physics: Conference Series. Retrieved March 8, 2022, from https://iopscience.iop.org/article/10.1088/1742-6596/1916/1/012039/pdf

International journal of innovation in engineering research and technology. Retrieved March 8, 2022, from https://repo.ijiert.org/index.php/ijiert/article/view/2569/2354

Deepak Birla | Social Distancing AI using Python, Deep Learning and Computer Vision.Retrieved March 8, 2022, from https://medium.com/@birla.deepak26/socialdistancing-ai-using-python-deep-learning-c26b20c9aa4c

PyImageSearch | Adrian Rosebrock, PhD. OpenCV Social Distancing Detector. Retrieved March 8, 2022, from https://pyimagesearch.com/2020/06/01/opencvsocial-distancing-detector/