Get the code: [https://github.com/nicknochnack/Multi...](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbG04Z3VrOHBpUnlWcEE0a1dfaDhVSlRIWkozd3xBQ3Jtc0ttQk5QOFdTaG1EVEswQVhfMDEweVhxX1BwdHdHdXdUakRydm05NlNMcS1pTTk5SzhiNE52TENudk1udHVwRXJkYlpEVlJ3cUI1c0pzVVpWc2hZSnFGUGlqSzM4akU3dlZDc2tVR18wekVXTVRUU3U4TQ&q=https%3A%2F%2Fgithub.com%2Fnicknochnack%2FMultiPoseMovenetLightning)

Links Model: [https://tfhub.dev/google/movenet/mult...](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbWhYNVlTSlZEdE9VSDlXQ3Q5RlNWYnpTQjBEQXxBQ3Jtc0tuYTlxYUJoUEI4ZGtNWEpLYzhVM1FCY1RUU1EyNXRmTmtiN3dvV2c5bTZ5Z0taeEluaEVnNnRMdEdaUkJheGF4UVgwNnJyVlNlS2ozVC1UN3FyVk1pUEZyMHZrS3QwckQ2VUhoLUJPMWdXalE1NC1iZw&q=https%3A%2F%2Ftfhub.dev%2Fgoogle%2Fmovenet%2Fmultipose%2Flightning%2F1)

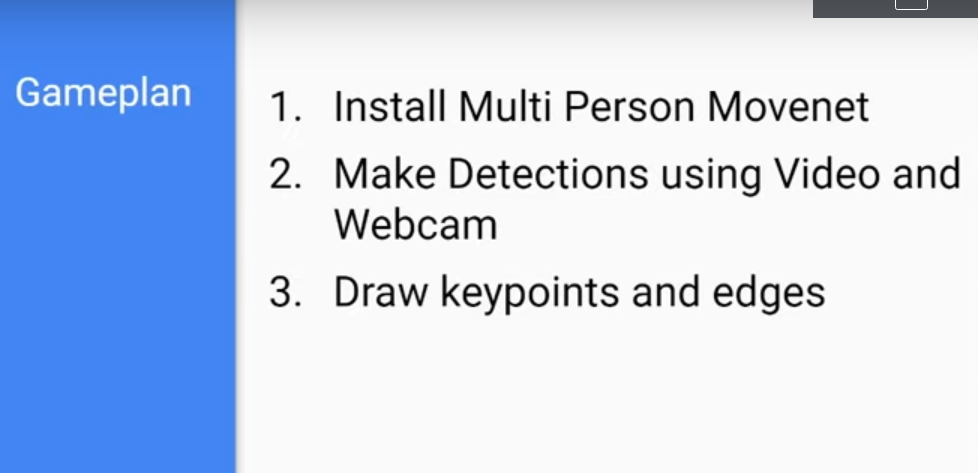
Existing Movenet Model: [https://github.com/nicknochnack/MoveN...](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbk5aUGlOLVVYRjBpZ1M4T2UtYzhZd3FXWllCQXxBQ3Jtc0tuYThtRFpzcWxXbS1JWnBrV0M3eThKamstSmlFc0JCMEwzYUIwYkx0djhnbVVOX1F3M2ZCSkJjVXk4MUlCQ0wtSWdudmU4Vl9NNUdrY2NzWUt0blVxN3lZelIzOGFGc3Q1MWFkZGtJUXJuWlZoM21LYw&q=https%3A%2F%2Fgithub.com%2Fnicknochnack%2FMoveNetLightning)

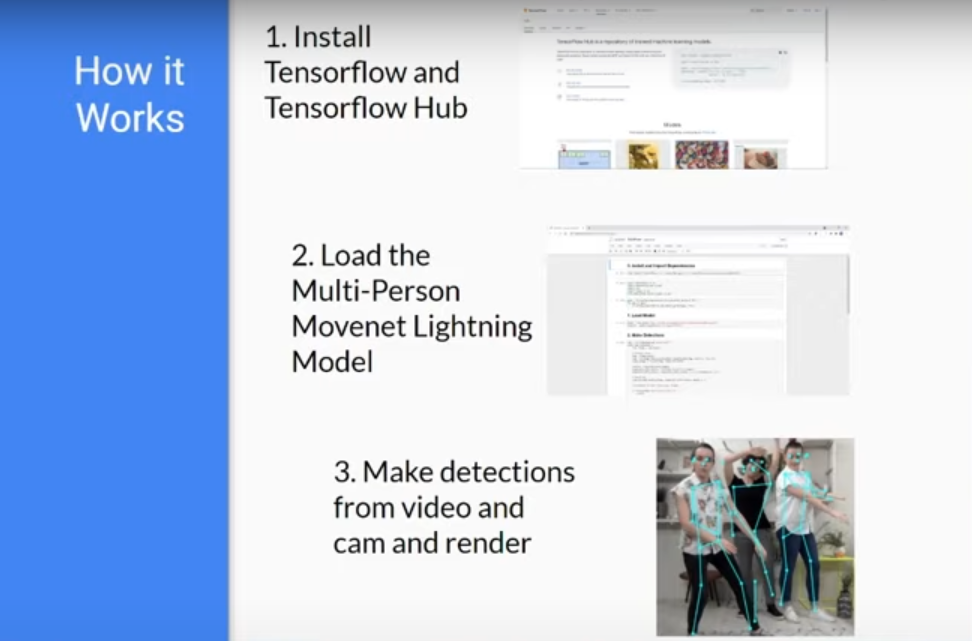
Papers

MobileNetv2: [https://arxiv.org/pdf/1801.04381.pdf](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqa19SUFFRMnZKekZPcUxmX1pvUUZFa3AwazQzd3xBQ3Jtc0tuRTZNeDN1VzVEVUQtTnhabTQ4clF3bEFDaVpuMmp2cVE1VUVXSlAtQ19oeTk5S1R1NTJweHdZRkdZamw5bnJfZlFyZVE5OWhFVGlTUlBlOWp3UVY0eHlMOUN2OTNDU1VzajZOaVEybFNPeDhaTmdySQ&q=https%3A%2F%2Farxiv.org%2Fpdf%2F1801.04381.pdf)

Feature Pyramid Networks: [https://arxiv.org/pdf/1612.03144.pdf](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbE8zVjlITUlZeGRHQVk4TW9hbE8taWdNRnRiQXxBQ3Jtc0trMGxZMEdvR2JoclJreDZHXzNqTkZZeTVlTXNQNGRXUXFJc2k3TFJJWV93TGg0ZnNmbmZjUUFWSnpnUFhoNFlSVklhX0ROcml1NG44M0RYTUxfWkNlRXRzZ2Fwb0pUV2FxdUlvVHBGa19va0tCOGZmTQ&q=https%3A%2F%2Farxiv.org%2Fpdf%2F1612.03144.pdf)

CenterNet: [https://arxiv.org/pdf/1904.07850.pdf](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbW1NSDh2VGpmM2VkQ0xYOFo1bWp5TlBiaGRXZ3xBQ3Jtc0ttWTBLM0xSMjJhT2pDWXFJNEdzbENZdjRSNHo3UUQxOEc0d3R1NkxnQmpHczdGWmc3M1pmOWlXWWhwU0lJQ1hhVlN2UWlpLVQ3cXR4Y0hzQzNjRWlfWjhBREFaUW1IZF91R1hTaTNOeW9XWGJEeDRoYw&q=https%3A%2F%2Farxiv.org%2Fpdf%2F1904.07850.pdf)





* Movenet/multipose/lightning 🡪 Image Pose detection
  + The model can detect up to 6 people in the image frame simultaneously
  + 17 key points per person
* Tensorflow hub helps in loading the multipose detection model
* Matplot to render image
* Numpy to help with drawing key points and edges
* Movenet expects:
  + - The height/width are both multiple of 32
* 2. Make Detections
* Resize
  + Make copy of frame and assign to variable **img**
  + Resize to multiple of 32 – 256/512/768 etc
  + Change the datatype to int32
* Detection
  + results = movenet(input\_img)
  + keypoints\_with\_scores = results['output\_0'].numpy()[:,:,:51].reshape**((6,17,3))**
  + # Render keypoints
  + loop\_through\_people(frame, keypoints\_with\_scores, EDGES, 0.1)
  + cv2\_imshow(frame)
* Extras
  + Results 🡪 shape=(1, 6, 56)
    - 1 – set of arrays
    - 6 people detected
    - 56 keypoints
    - Y,x,score \* 17 = 51 values
    - Last 5 is for bounding box coordinates for each
  + Apply the Transformation with keypoint scores
* results['output\_0'].numpy()[:,:,:51]
  + - re shape to 6,17,3
      * 6 people
      * 17 keypoints
      * 3 key values for each keypoints
* Draw Keypoints
* *def* draw\_keypoints(*frame*, *keypoints*, *confidence\_threshold*):
* y, x, c = frame.shape
* shaped = np.squeeze(np.multiply(keypoints, [y,x,1]))
* for kp in shaped:
* ky, kx, kp\_conf = kp
* if kp\_conf > confidence\_threshold:
* cv2.circle(frame, (int(kx), int(ky)), 6, (0,255,0), -1)

Keypoints are for 1 person and will be looped through for multiple persons

* 4. Draw Edges
* 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'
* *def* draw\_connections(*frame*, *keypoints*, *edges*, *confidence\_threshold*):
* y, x, c = frame.shape
* shaped = np.squeeze(np.multiply(keypoints, [y,x,1]))
* for edge, color in edges.items():
* p1, p2 = edge
* y1, x1, c1 = shaped[p1]
* y2, x2, c2 = shaped[p2]
* if (c1 > confidence\_threshold) & (c2 > confidence\_threshold):
* cv2.line(frame, (int(x1), int(y1)), (int(x2), int(y2)), (0,0,255), 4)
* Draw connection using edge coordinates 0 represents right eye
* Use draw\_keypoints & draw\_connections to render the results
* Function to loop through each person detected and render

# Function to loop through each person detected and render

*def* loop\_through\_people(*frame*, *keypoints\_with\_scores*, *edges*, *confidence\_threshold*):

    for person in keypoints\_with\_scores:

        draw\_connections(frame, person, edges, confidence\_threshold)

        draw\_keypoints(frame, person, confidence\_threshold)

* Render keypoints

**loop\_through\_people(frame, keypoints\_with\_scores, EDGES, 0.3)**

* Aspect ration for frame size:
  + Frame.shape 🡪 Frame size w\*h
  + w/h
  + 256\*(w/h)
  + Multiple of 32

Source: <https://www.youtube.com/watch?v=KC7nJtBHBqg>

Colab : <https://colab.research.google.com/drive/1NlPEUZG1kMrUDQAIdxD2XiNnyz_HY7zS?authuser=1#scrollTo=53cgPFMysP8i>