**Open Pose Multi Person Key Point Detection**

Prudhvi Kishan, Sri Ranga

Department of Computer Science, The University of Akron

**Abstract**

*The project’s objective is to detect key points in multiple people present in a single image using the Open Pose toolkit. The report starts by explaining about Open Pose, key point detection, various application areas and limitations. The report also deals with the implementation of the project using Python and libraries like Coco and MPI.*

**1. Introduction**

Human key point estimation has been an open problem for decades in the research community. Initially, efforts were focused on facial alignment (i.e., face key point detection) [2, 3, 4, 5, 6, 7]. Gradually, the problem evolved into single and multi-person human pose estimation in-the-wild, including body and foot key points [8, 9, 10, 11, 12]. A more recent and challenging problem has targeted hand key point detection [13, 14, 15]. The next logical step is the integration of all of these key point detection tasks within the same algorithm, leading to “whole-body” or “full-body” (body, face, hand, and foot) pose estimation [16, 1].

There are several applications that can immediately take advantage of whole-body key point detection, such as re-targeting and 3D human key point and mesh reconstruction [17, 18, 19, 20, 21]. In general, almost any method that uses body information could also benefit from face, hand, and foot detection, such as person re-identification, tracking, or action recognition [22, 23, 24, 25]. Despite these needs, the only existing method providing whole-body pose estimation is the original Open Pose [1], which follows a multi-stage approach. It first obtains all body poses from an input image in a bottom-up fashion [10] and then runs additional face and hand key point detectors [13] for each detected person. As a multi-network approach, it directly uses the existing body, face, and hand key point detection algorithms. However, it suffers from early commitment: if the body-only detector fails, there is no recourse to recovery, and it is prone to do so when only a face or a hand are visible in the image. In addition, its runtime is proportional to the number of people in the image, making whole-body Open Pose prohibitively costly for multi-person and real-time applications. A single-stage method, estimating whole-body parts of multiple people in a single inference, would be more attractive as it would yield a fixed inference runtime, independent to the number of people in the scene.

**2. Background**

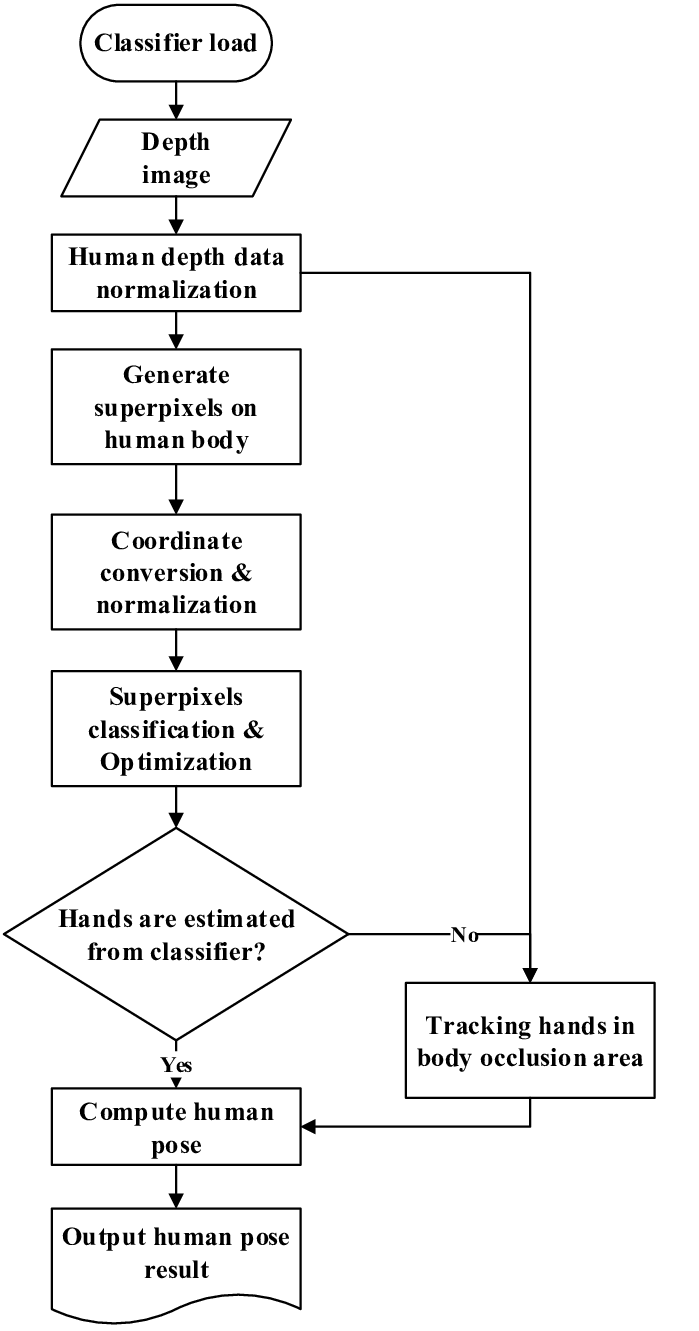
**2.1 Open pose**

Human pose estimation is the process of estimating the configuration of the body (pose) from a single, typically monocular, image. Background. Human pose estimation is one of the key problems in computer vision that has been studied for well over 15 years.

The process usually involves the extraction of joints on a human body, and then analysis of a human pose by using deep learning algorithms. If the human poseestimation system uses video records as a data source, key points (joints locations) are detected from a sequence of frames, not a single picture.

**3. Design**

**3.1 Flow Diagram**



The step by step division of the human body pose into multiple key points is done separately in each region of the body. The algorithm does Face Keypoint Detection, Body Keypoint Estimation, Foot Keypoint Estimation, Hand Keypoint Detection, Whole-Body Keypoint Detection, Multi-Task Learning and PAF-based Body Pose Estimation.

**3.2 Implementation**

**3.2.1 Models**

Caffe is a deep learning framework developed by the Berkeley Vision and Learning Center (BVLC). It is written in C++ and has Python and MATLAB bindings.

There are 4 steps in training a CNN using Caffe:

* Step 1 - Data preparation: In this step, we clean the images and store them in a format that can be used by Caffe. We will write a Python script that will handle both image pre-processing and storage.
* Step 2 - Model definition: In this step, we choose a CNN architecture and we define its parameters in a configuration file with extension .prototxt.
* Step 3 - Solver definition: The solver is responsible for model optimization. We define the solver parameters in a configuration file with extension .prototxt.
* Step 4 - Model training: We train the model by executing one Caffe command from the terminal. After training the model, we will get the trained model in a file with extension .caffemodel.

After the training phase, we will use the .caffemodel trained model to make predictions of new unseen data. We will write a Python script to this.

We downloaded the trained .caffemodel and .prototxt files separately for coco and mpi processes to use the model in our Python code.

**3.2.2 Algorithm Implementation**

The code is written in Python by using the python open source libraries such as OpenCV and NumPy. The source image is read as a default argument in the code. Time library is used to calculate the time required for the key point detection to be done and to get the human pose output. The prototxt and caffemodel are called in the code and are stored as *protoFile* and *weightsFile*. Keypoint mapping dictionary is initialized in the code. The dictionary contains all possible separation of each and every region of the body like Nose, Neck, Shoulder, Elbow, Hip, Eye etc. Pose pairs, mapIds and color coding dictionaries are created.

Keypoints are obtained by using the *Gaussian Blur* method and contours are estimated.

Person wise Keypoints are calculated in the single image and are merged in pairs by taking only the valid pairs.

There are two main steps in the implementation of the key point detection of multiple people in the image.

Confidence Maps is used for 2D presentation of the particular body part in any given pixel.

Part Affinity is a set of 2D vector fields that encodes location and orientation of limbs of different people in the image.

**4. Results**

A picture containing athletic game, sport, outdoor, person

Description automatically generated A group of people wearing clothing

Description automatically generatedA picture containing text, indoor

Description automatically generated

**5. Observations**

There are few limitations in using Openpose for keypoint detection of multiple people in the image. A few limitations are listed below:

* Current human pose performance metrics are based on key point accuracy.
* Caffe models are generated using the CNN algorithm. There is a slight trade-off between speed and accuracy. R-CNN algorithms runs faster
* There is no completely fair comparison
* Failure cases still exist (i.e. foot and leg occluded, rare joint position, etc.)
* Scales well to GPU over CPU

The keypoint detection runs faster if we replace the Message Passing Interface(MPI) with that of any GPU based system.

**6. Conclusion & Applications**

The use of non-parametric color-coded PAFs creates greater accuracy for mapping.

High accuracy is achieved without compromising on execution performance.

Openpose is a evergreen technology in the field of computer vision.

Analogous to what Fast R-CNN did for object detection, our work brings together multiple and, currently, independent keypoint detection tasks into a unified framework. We evaluate our method on multiple keypoint detection benchmarks and compare it to the state-of-the-art (our previous work, OpenPose), considerably outperforming it in both training and testing speed as well as slightly improving its accuracy.

Nevertheless, there are still some limitations with our method.

First, we observe global failure cases when a significant part of the target person is occluded or outside of the image boundaries.

Secondly, the accuracy of the face and especially hand keypoint detectors is still limited, recursively failing in the case of severe motion blur, small people, and extreme gestures.

OpenPose crops the bounding box proposal of those bounding box candidates, resizes them up, and feeds them into its dedicated networks. This higher input resolution leads to an increased pixel localization precision if the keypoint detection is successful.

We can conclude that pose estimation approaches are approaching the state-of-the-art in computer vision. These methods have concrete applications in industry and study. However, due to occlusion of joints and anomalous angles, it will be a while before a practically perfect pose estimation implementation.

**6.1 Application Areas of the Pose Estimation**

1. Assisted Living
2. Character Animation
3. Video Games
4. Intelligent Driver Assist System
5. Medical Applications
6. Besides these, the different applications of pose estimation include video surveillance, behaviour understanding, sign language detection, markerless motion capturing, human-computer interaction, and animal tracking.

**7. Future Work**

* A calibrated, multi-view imaging setup is needed to maximize the estimation accuracy. For this reason, a set of cameras will be placed in precise locations and will be calibrated, in order to better account for the effects of varying scale due to lens distortion. These cameras will be used to simultaneously capture different views of the body pose.
* The model preparation, code execution and the image pixel division can be done more accurately and precisely on a GPU device.
* Human Activity Recognition has wide range of applications in multiple domains. It becomes easy to use when working with the images rather than installing sensors all over the body. Existing system can be integrated with hand and facial gesture recognition. By identifying unique person identity in an image, it can recognize multiple human activities. It is developed with TensorFlow python Application Program Interface. Caffe installation can improve this result up to 25-30 frames per second.

**8. References**

* Maithani, M. (2021, January 12). *Guide to OpenPose for Real-time Human Pose Estimation*. Analytics India Magazine. <https://analyticsindiamag.com/guide-to-openpose-for-real-time-human-pose-estimation/>
* Tanugraha, P. (2019, September 20). *Understanding OpenPose (with code reference)— Part 1*. Medium. <https://medium.com/analytics-vidhya/understanding-openpose-with-code-reference-part-1-b515ba0bbc73>
* Sharma, S. (2020, April 6). *A Short Guide to Pose Estimation in Computer Vision*. Medium. <https://medium.com/@siddrrsh/a-short-guide-to-pose-estimation-in-computer-vision-3ea708dd9155>
* Kanjee, R. (2019, May 3). *What You Need to Know About Pose Estimation and OpenPose?* Medium. <https://augmentedstartups.medium.com/what-you-need-to-know-about-pose-estimation-4ac189a5e07e>
* Odemakinde, E. (2021, April 16). *State of the Art Pose Estimation: The Ultimate Overview*. Viso.Ai. <https://viso.ai/deep-learning/pose-estimation-ultimate-overview/>
* Mwiti, D. (2021, April 9). *A 2019 Guide to Human Pose Estimation - Heartbeat*. Medium. <https://heartbeat.fritz.ai/a-2019-guide-to-human-pose-estimation-c10b79b64b73>