# Perform Pose Estimation using Computer Vision

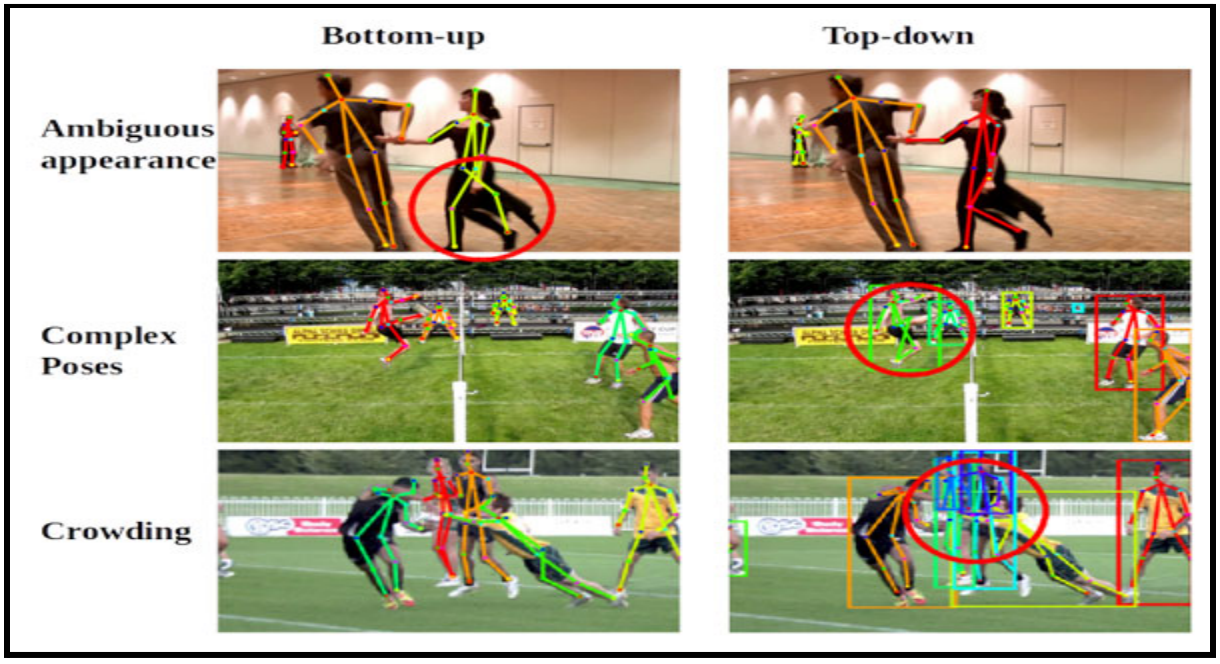
## What is Pose Estimation?

Pose estimation is a computer vision technique that predicts and tracks the location of a person or object. This is done by looking at a combination of the pose and the orientation of a given person/object.
This is typically done by identifying, locating, and tracking a number of keypoints on a given object or person. For objects, this could be corners or other significant features. And for humans, these key points represent major joints like an elbow or knee.

## Primary Techniques for Pose Estimation

In general, deep learning architectures suitable for pose estimation are based on variations of convolutional neural networks (CNNs). For a gentle introduction, check out this guide to convolutional neural networks.

There are two overarching approaches: a bottom-up approach, and a top-down approach.
With a bottom-up approach, the model detects every instance of a particular keypoint (e.g. all left hands) in a given image and then attempts to assemble groups of keypoints into skeletons for distinct objects.
A top-down approach is the inverse – the network first uses an object detector to draw a box around each instance of an object, and then estimates the key points within each cropped region.



and though we referenced this previously, it's important to remember the distinction between 2D pose estimation and 3D pose estimation. The two tasks carry different data requirements, produce different outputs (2D pixel values vs a 3D spatial arrangement), and are generally used to solve different problems

## Basic structure

Deep learning models for pose estimation come in a few varieties related to the top-down and bottom-up approaches discussed above. Most begin with an encoder that accepts an image as input and extracts features using a series of narrowing convolution blocks. What comes after the encoder depends on the method of pose estimation.

The most conceptually simple method uses a regressor to output final predictions of each keypoint location. The resulting model accepts an image as input and outputs X, Y, and potentially Z coordinates for each keypoint you're trying to predict. In practice, however, this architecture does not produce accurate results without additional refinement.

A slightly more complicated approach uses an encoder-decoder architecture. Instead of estimating keypoint coordinates directly, the encoder is fed into a decoder, which creates heatmaps representing the likelihood that a keypoint is found in a given region of an image.

During post-processing, the exact coordinates of a keypoint are found by selecting heatmap locations with the highest keypoint likelihood. In the case of multi-pose estimation, a heatmap may contain multiple areas of high keypoint likelihood (e.g. multiple right hands in an image). In these cases, additional post-processing is required to assign each area to a specific object instance.

Top-down approaches also use an encoder-decoder architecture to output heatmaps, but they contain an additional step. An object detection module is placed between the encoder and decoder and is used to crop regions of an image likely to contain an object. Keypoint heatmaps are then predicted individually for each box. Rather than having a single heatmap containing the likely location of all left hands in an image, we get a series of bounding boxes that should only contain a single keypoint of each type. This approach makes it easy to assign key points to specific instances without a lot of post-processing.

Both types of architectures discussed thus far apply equally to 2D and 3D pose estimation. There are, however, a few options worth mentioning that are specific to 3D

rigid pose estimation. Rather than predicting the coordinates of each keypoint in [X, Y, Z] space, it's instead possible to predict the 6 degrees of freedom of an object and apply the transformation to a reference skeleton. For example, instead of trying to predict the 3D position of the top of a water bottle, you would instead predict the 3D coordinates of it's center of mass and then predict the rotation of the bottle in an image relative to a reference coordinate system. This method of prediction is particularly useful for applications in AR, VR, or other apps that use 3D rendering software like Unity. It can, however, be hard to train accurate models.

References:


- https://www.fritz.ai/pose-estimation/#part-basics
- https://www.ibm.com/topics/computer-vision