AI Powered Gym Trainer

Venkatakrishnan R
CH.EN.U4AIE20078
Computer Science and Engineering (AI)
Amrita Vishwa Vidyapeetham, Chennai

Siva Jyothi Nath Reddy B
CH.EN.U4AIE20063
Computer Science and Engineering (AI)
Amrita Vishwa Vidyapeetham, Chennai

Sarthak Yadav
CH.EN.U4AIE20058
Computer Science and Engineering (AI)
Amrita Vishwa Vidyapeetham, Chennai

Shaik Huziafa Fazil
CH.EN.U4AIE20060
Computer Science and Engineering (AI)
Amrita Vishwa Vidyapeetham, Chennai

Pravin Mukesh
CH.EN.U4AIE20050
Computer Science and Engineering (AI)
Amrita Vishwa Vidyapeetham, Chennai

I. Introduction

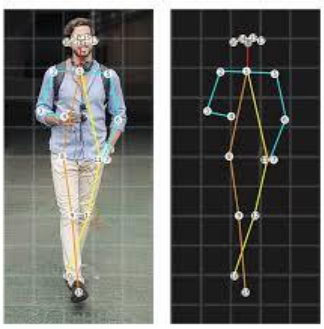
Pose estimation is a computer vision task that infers a person's or object's pose from a photograph or video. Pose estimation can alternatively be defined as the challenge of determining the location and orientation of a camera in relation to a person or an item.

This is usually accomplished by locating, tracking, and identifying a number of keypoints on a specific object or person. This could be corners or other distinguishing elements on an object. These keypoints represent significant joints in humans, such as the elbow and knee.

Our machine learning algorithms are tasked with detecting these keypoints in photos and videos.

II. Catagories of pose estimation

The goal is to track down the keypoints in the video.



These keypoints indicate important joints such as elbows, knees, wrists, and so on while working with individuals. Human pose estimate is the term for this. Humans belong to a category of items that are adaptable. Keypoints will be in different places compared to others if we bend our arms or legs. The majority of inanimate objects are inflexible. For example, regardless of the orientation of a brick, its corners are always the same distance apart. Rigid pose estimation is the process of predicting the position of these objects.

A distinction must also be established between 2D and 3D pose estimate. The placement of keypoints in 2D space relative to an image or video frame is readily estimated with 2D pose estimation. For each keypoint, the model calculates an X and Y coordinate. By adding a z-dimension to the forecast, 3D posture estimation transforms an object in a 2D image into a 3D object.

We can forecast the true spatial placement of a displayed person or item using 3D pose estimation. Given the complexity necessary in building datasets and algorithms that take into consideration a range of parameters – such as an image's or video's background scene, lighting conditions, and more – 3D pose estimation is a more difficult task for machine learners.

Finally, there is a distinction to be made between detecting one thing in an image or video and detecting numerous objects in an image or video. Single and multi pose estimation are the names for these two methodologies, which are essentially self-explanatory: Multi pose estimation approaches detect and track many persons or objects, whereas single pose estimation approaches detect and track one person or item

III. WHY DOES POSE ETIMATION MATTER?

We can monitor an object or person (or numerous individuals) in real-world space at an extraordinarily detailed level using posture estimation. This tremendous capacity brings up a plethora of potential uses.

In some important aspects, pose estimation varies from other standard computer vision tasks. Object detection is a task that locates objects inside an image. However, this localization is usually coarse, consisting of a bounding box that encompasses the object. Pose estimate goes much further, estimating the exact location of the object's keypoints.

When we analyse how pose estimate may be used to automatically detect human movement, we can see how powerful it is. Pose estimation has the potential to develop a new wave of automated systems meant to quantify the precision of human movement, from virtual sports coaches and AI-powered personal trainers to tracking movements on factory floors to ensure worker safety.

Pose estimation, in addition to tracking human movement and activity, has a wide range of applications, including:

- Augmented Reality
- Animation
- Game
- Robotics

IV. K NEAREST NEIGHBOUR

The KNN algorithm believes that objects that are similar are close together. To put it another way, related items are close together.

There are numerous methods for determining distance, and depending on the task at hand, one method may be preferred. The straight-line distance (also known as the Euclidean distance) is a common and well-known option.

<u>The KNN Algorithm :</u>

```
1. Load the data
2. Set K to the number of neighbours you want.
3. For each example in data :
    I. From the data, calculate the distance
       between the query example and the
       current example.
```

II. To an ordered collection, add the example's distance and index.

4. Sort the ordered set of distances and indices by their distances from least to greatest (in ascending order).
5. Choose the first K entries from the sorted list.
6. Get the labels for the K entries you've chosen.
7. Return the mean of the K labels if there is a regression.
8. Return the mode of the K labels if there is a classification.

Choosing the right value of K :

We run the KNN algorithm numerous times with different values of K to find the K that decreases the amount of errors we encounter while retaining the algorithm's capacity to generate correct predictions when it's given data it hasn't seen before.

Here are a few things to remember:

1. Our forecasts become less stable as we reduce the value of K to one. Consider the case where K=1 and the query point is surrounded by numerous reds and one green (I'm thinking of the top left corner of the coloured plot above), but the green is the lone closest neighbour. We would expect the query point to be red, but because K=1, KNN wrongly predicts that the query point would be green.
2. Inversely, as the value of K grows larger, our forecasts become more stable due to majority voting / averaging, and hence more likely to be accurate (up to a certain point). We eventually start to see an increase in the amount of errors. We know we've pushed the value of K too much at this point.
3. We commonly make K an odd number to have a tiebreaker in circumstances where we take a majority vote among labels (e.g. determining the mode in a classification problem)

Advantages :

1. The algorithm is straightforward and simple to implement.
2. There's no need to create a model, tweak a few parameters, or make any more assumptions.
3. The algorithm is extremely adaptable. It has classification, regression, and search capabilities (as we will see in the next section).

Disadvantage :

1. As the number of samples and/or predictors/independent variables grows, the process becomes much slower.

KNN in Practice :

The fundamental downside of KNN is that it becomes much slower as the volume of input grows, making it an unsuitable solution in situations when rapid predictions are required. Furthermore, quicker algorithms can offer more precise classification and regression results.

KNN, on the other hand, can be beneficial in addressing issues whose solutions rely on finding similar items if you have enough computational resources to handle the data you're utilising to generate predictions quickly. One use of KNN-search is the use of the KNN algorithm in recommender systems.

V. MediaPipe

MediaPipe is a framework for creating applied machine learning pipelines that are multimodal (e.g. video, audio, any time series data) and cross platform (i.e. Android, iOS, web, edge devices). A perception pipeline can be constructed using MediaPipe as a network of modular components, such as inference models (e.g., TensorFlow, TFLite) and media processing functions.

Cutting Edge ML Model :

- Face Detection
- Multi – hand Tracking
- Hair Segmentation
- Object Detection
- Objectron : 3D Object Detection and Tracking
- AutoFlip : Automatic video Cropping Pipeline

Nest, Gmail, Lens, Maps, Android Auto, Photos, Google Home, and YouTube are just a few of the Google products and teams that use MediaPipe.

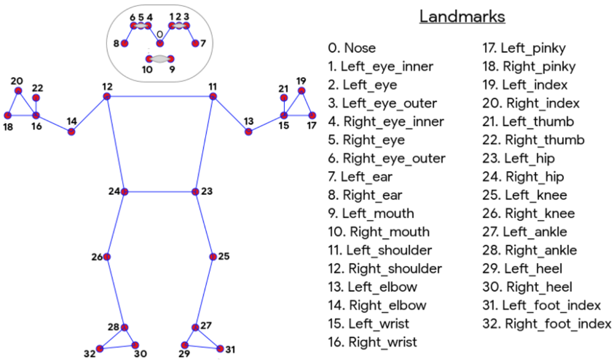
VI. BlazePose

Google's BlazePose (Full Body) posture detection model can compute the (x,y,z) coordinates of 33 skeleton keypoints. It can be utilised in fitness applications, for example.

There are two machine learning models in BlazePose: a Detector and an Estimator. The Detector removes the human region from the input image, while the Estimator inputs a 256x256 resolution image of the discovered person and returns keypoints.

The 33 keypoints are output by BlazePose according to the following ordering convention. This is more than the COCO dataset's 17 keypoints, which are typically used.

Note : The MS COCO dataset (Microsoft Common Objects in Context) is a large-scale dataset for object detection, segmentation, key-point detection, and captioning. There are 328K photos in the dataset.



Architecture :

The Detector's architecture is based on Single-Shot Detectors (SSD). It returns a bounding box and a confidence score after receiving an input image. (x,y,w,h,kp1x,kp1y,...,kp4x,kp4y) are the 12 elements of the bounding box, where kp1x to kp4y are additional keypoints. Each of the 2254 pieces need its own anchor, anchor scale, and offset.

You can utilise the Detector in two ways. The bounding box in box mode is calculated using the position (x,y) and size of the bounding box (w,h). The scale and angle of (kp1x,kp1y) and (kp2x,kp2y) are calculated in alignment mode, and the bounding box, including rotation, may be projected.

For faster inference, the Estimator employs heatmap for training but computes keypoints directly without using heatmap.

The Estimator's first output is landmarks, and its second output is (1,1) flags. The landmarks are made up of 165 pieces for each of the 33 keypoints (x,y,z, visibility, presence).

When the z-value is negative, keypoints are between the hips and the camera; when the value is positive, keypoints are behind the hips.

The visibility and presence values are saved in the [min float,max float] range and transformed to probability using a sigmoid function. The visibility property returns the likelihood of keypoints in the frame that are not obscured by other objects. The probability of keypoints in the frame is returned by presence.

Usage :

To execute BlazePose (Full Body) with the ailia SDK, type the following command.

Only the upper body may be estimated using the BlazePose (Upper Body). At first, MediaPipe only released the upper body model, then the entire body model. The complete body and upper body versions have distinct specs; for example, the upper body model's detector resolution is 128x128.

VII. Applications of Pose Estimation

We'll go over several real-world use cases for pose estimation in this section. We've touched on a few of them in earlier sections, but in this area, we'll delve a little deeper and look at how this computer vision approach can be applied across industries.

We'll look at how pose estimation is applied in the following areas:

A. *Human Activity and Mavement :*

Pose estimation can be used to track and measure human movement, which is one of the most obvious applications. However, tracking movement isn't something that can be put into production in and of itself. The applications that come from tracking this movement, however, are dynamic and far-reaching with a little creative thinking.

Consider an AI-powered personal trainer that operates by simply pointing a camera at a person conducting a workout and letting a human pose estimation model (trained on a number of specific poses relevant to a training regimen) determine whether or not a certain exercise has been executed correctly.

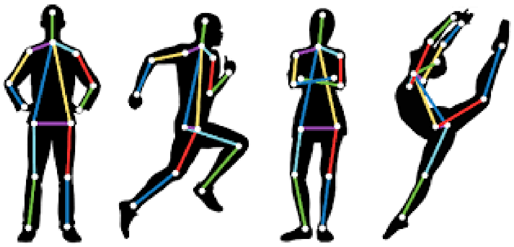


Figure 1. Example human pose trees

This type of app could make home fitness programmes safer and more inspiring, while simultaneously boosting accessibility and lowering the costs associated with professional physical trainers.

And, because pose estimation models can now run on mobile devices without requiring internet access, this type of application might readily expand access to this type of expertise to rural or otherwise difficult-to-reach locations.

Other experiences that could be powered by using human pose estimation to track human movement include, but are not limited to:

1. AI Powered Sports Coaches
2. Workplace Activity Monitoring
3. Crowd Counting and Tracking

B. *Augemented Reality and Experiences :*

Pose estimation, while not immediately apparent, offers the potential to develop more realistic and responsive augmented reality (AR) experiences.

We've spent a lot of time in this article talking about human posture estimation. However, as you may recall from Part 1 of this article, we can also use non-variable keypoints to locate and track objects. Rigid posture estimation allows us to determine a particular object's principal keypoints and track them as they move across real-world locations, from pieces of paper to musical instruments to...well, pretty much everything you can think of.

What does this have to do with AR? In essence, augmented reality allows us to place digital elements in real-world settings. This could be trying on a pair of digitally produced shoes or testing out a piece of furniture in your living room by placing a 3D rendering of it in the space.

So, where does posture estimate come into play? If we can reliably find and monitor a physical object in real-world space, we can overlay a digital augmented reality object onto the tracked thing.

The metaverse is a hybrid of virtual, augmented, and physical reality that blurs the lines between online and offline activities. But, to put it another way, it's a collection of platforms like the Sandbox, Mirandus, and Decentraland where people can communicate in a variety of ways. Since Mark Zuckerberg declared that Facebook would change its name to Meta and spend at least $10 billion on the metaverse, interest in it has exploded. Businesses have already begun to start new ventures in this digital area as more people continue to make their bets on a future embedded in the metaverse.

C. *Animation and Gaming :*

Character animation has traditionally been a labor-intensive technique that relies on large, expensive motion capture devices. However, with the development of deep learning approaches to pose estimation, these systems have the potential to be streamlined and automated in many ways.

This transition is achievable thanks to recent developments in pose estimation and motion capture technologies, which allow for character animation without the use of markers or specialist suits while still capturing motion in real time.

Similarly, deep learning-based pose estimation has the ability to automate the capture of animations for immersive video gaming experiences. Microsoft's Kinect depth camera popularised this type of gaming experience, and developments in gesture recognition promise to meet the real-time requirements that these systems demand.

D. *Robotics :*

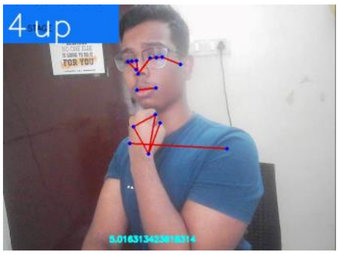
Traditionally, 2D vision systems have been used in industrial robotics to enable robots to accomplish their varied jobs. This 2D technique, however, has a number of drawbacks. Computing the point to which a robot should travel given this 2D representation of space, for example, necessitates extensive calibration operations and, unless reprogrammed, becomes inflexible to environmental changes.

However, with the introduction of 3D pose estimation, the possibility of creating more responsive, adaptable, and precise robotics systems now exists.

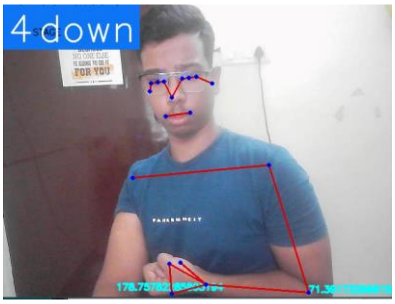
VIII. Results

The algorithm counts the number of upstrokes and down stroke performed while biceps curls.

Up :



Down :



IX. Conclusion

Pose estimation is an intriguing component of computer vision that has applications in a variety of industries, including technology, healthcare, and business. It is also utilised for security and surveillance systems, in addition to modelling human personalities using Deep Neural Networks that learn numerous vital points.

References

https://openaccess.thecvf.com/content_cvpr_2014/papers/Toshev_DeepPose_Human_Pose_2014_CVPR_paper.pdf

https://medium.com/axinc-ai/blazepose-a-3d-pose-estimation-model-d8689d06b7c4

https://google.github.io/mediapipe/solutions/pose.html