Report “Human pose estimation”

Group: IT-2204
Nauruzova Anel, Korfildesova Saltanat, Bekbolat Zhanerke

1. Introduction:

1.1. Problem

Human Pose Estimation (HPE) is one of the most important tasks among computer vision tasks, where the configuration of body parts of an object is determined either from images or videos. HPE has found wide application in the areas of augmented reality, animation, athlete performance analysis and surveillance. The problem is related to the variety of human poses, the occlusion of some body parts, the wide variety of environmental conditions and different camera points of view.

1.2. Literature review with links (another solutions)

In the very recent past, the growth of human pose estimation (HPE) has been colossal, majorly attributed to deep learning technologies and their furthering with computer vision technologies. This part shall briefly go through some of the most seminal and recent works in the field that have developed the same, touching on various methodologies and approaches that have been used to do so.

The first analyzed article in this study is: "3D Human Pose Estimation in Video with Temporal Convolutions and Semi-supervised Training" [1]. This work relates to the human poses featured in video sequences. They designed a temporal model using deep networks, which allows for better action recognition while capturing spatiotemporal motion information effectively over time. They further came up with the ideas of semi-supervised training in a manner of using both labeled and unlabeled data, giving its way for better performance. The improvement is particularly important for this particular type of method over previous works for dynamic poses and movements captured from complex video sequences. Authors in their work show the datasets Human3.6M and HumanEva-I. Human3.6M has 3.6 million video frames recorded across 11 subjects, out of which seven are annotated with 3D poses. In every video, 15 actions per sequence are carried out, captured from the four synchronized cameras running at 50 Hz.

The second work is “OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields”[2]. This work is something like our goal,

but it is more about measuring the body parts of humans—parts like the head, shoulders, arms, etc. It then introduces Part Affinity Fields (PAFs) in capturing the part-pair relationship, such that in the end, effective estimation of poses of multiple persons in real time can be made. This OpenPose has been adopted for the guarantee of robustness and flexibility, even in extremely busy contexts where people are involved, and it has reached fields such as animation and sports analysis. They use an MPII human multi-person dataset that consists of 3844 training and 1758 testing group images of many interacting persons in highly articulated poses with 14 body parts. Another two datasets used are the COCO keypoint challenge dataset, which can find 17 keypoints of the human body, and their own foot dataset.

The third one is the work "Deep High-Resolution Representation Learning for Human Pose Estimation". Their main goal is that the proposed network maintains high-resolution representations through the whole process. The authors propose a new architecture, called High-Resolution Network (HRNet), to achieve their goal, as mentioned before. The main idea is to keep the high-resolution features across the whole network, in contrast to classical downsampling of the image. These add significantly: the accuracy of pose estimation better than the existing approaches in distinguishing the closeness of body parts and dealing with small-scale body parts of the body from a highly resolved image. They also made use of some datasets like COCO, MPII Human Pose Estimation, Application to Pose Tracking (PoseTrack dataset).

Next is "Simple Baselines for Human Pose Estimation and Tracking" [4]. In comparison with the most recent frameworks for human pose estimation and tracking, usually featuring very large and complex models, simpler and well-organized architectures have the potential of showing very competitive results. In a very simple way, the authors have suggested quite a reasonable, efficient, and effective baseline model, so it can be very handy in the process of training and implementation. Involvement of the convolutional network components in a very simple model, but of a great power of performance over various datasets and scenarios. These are simple and efficient baselines that not only provide good reference points for the community, but suggest that these can well be advanced also through optimization and further refinement, rather than further complexity.

The last work is "CMU Panoptic Studio: A Massively Multiview System for Social Interaction Capture" [5]. If the focus has been on primary considerations toward capturing social interactions, then CMU Panoptic Studio

has taken a long leap in the field of human pose estimation. More than 500 geodesic dome-arranged cameras allowed capturing social interaction from any possible angle. This leads to a quite rich data set of human poses and movements in detail and with variety like never before. Data produced by the system were used further for the development and fine-tuning of pose estimation models under complex scenarios with interaction of more than one person. The Panoptic Studio is first of its kind in the world, and there is nothing like it in the push of bounds of multi-person pose estimation, therefore gaining fundamental insights into human behavior and dynamics. Each of them contributed unique value to the field of human pose estimation, solving some challenges along the way and, all together, pushing the field toward more precision, efficiency, and dexterity.

Each of these works contributes uniquely to the advancement of human pose estimation, addressing specific challenges and pushing the field towards more accurate, efficient, and versatile solutions. The continuous innovation reflected in these studies underscores the dynamic nature of research in computer vision and machine learning, with each advancement building on the foundations laid by previous work.

1.3. Current work (description of the work)

In this project, we employ the *MediaPipe* framework integrated with *OpenCV* to perform human pose estimation tasks. *Our aim* is to detect and analyze human poses from real-time image snapshots using advanced machine learning models provided by MediaPipe. This technology allows for real-time, efficient, and accurate pose estimation which is crucial for various applications such as fitness coaching, animation, augmented reality, and surveillance.

Data Collection and Preprocessing:

The images used for pose estimation are taken from the MPII Human Pose dataset, which includes various people, backgrounds, and lighting conditions to ensure the robustness of our pose estimation model. These images are preprocessed and fed into the MediaPipe Pose pipeline. The preprocessing steps include scaling, normalization, and conversion from BGR to RGB color space as required by the model.

Pose Estimation Model:

We utilize the MediaPipe Pose solution with a model complexity level of 2, which provides a balance between accuracy and computational efficiency. This level is selected based on the requirement for high precision in pose

detection while maintaining real-time performance. The pose estimation process involves detecting 33 unique landmarks on the human body, including key points on the face, torso, arms, and legs.

Implementation:

The implementation is conducted using Python (Google Colab), leveraging libraries such as MediaPipe for pose estimation and OpenCV for image processing tasks. The workflow involves reading images from a designated path, converting the images into RGB format for compatibility with the MediaPipe model, and then applying the pose detection algorithm to identify human body landmarks.

We have also developed a series of functions to facilitate the visualization and analysis of the detected poses:

1. Pose Detection in Static Images: A function detect_pose_in_image() that takes an image path as input, processes the image, and returns pose landmarks detected by MediaPipe.
2. Visualization: Utilizing Matplotlib and OpenCV to plot the detected landmarks and connections on the human body, providing intuitive visual feedback on the pose estimation results.
3. Pose Classification: Custom functions like classifyPose() to analyze detected landmarks and classify the pose into predefined categories such as sitting, standing, or lying based on the angles and positions of key body parts.
4. 3D Pose Visualization: Extending beyond 2D pose estimation, we explore the potential for 3D visualization of detected poses, enhancing the depth and utility of pose analysis, especially for applications requiring spatial understanding of body posture.

Challenges and Solutions:

During the implementation, we encountered challenges related to varying lighting conditions, occlusions, and complex human poses. To address these, we enhanced the dataset diversity and refined the model parameters to improve detection robustness. Furthermore, we implemented error-handling mechanisms to gracefully manage cases where pose detection is not feasible.

Tools and Technologies:

- MediaPipe: For leveraging state-of-the-art pose estimation models.
- OpenCV: For image manipulation and preprocessing.

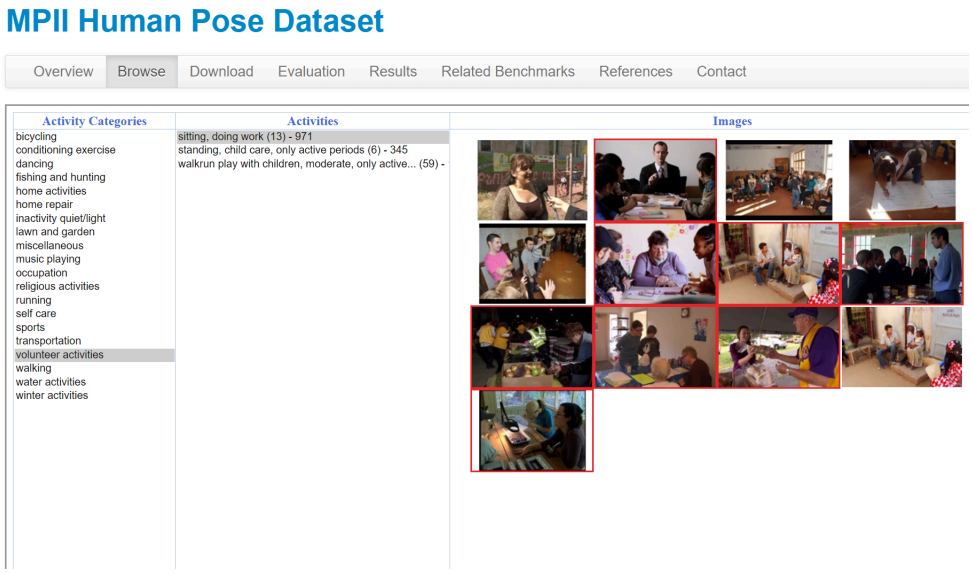
- Python: As the primary programming language for scripting and model integration.
- Matplotlib: For plotting and visualizing pose landmarks and angles.

In summary, our current work involves the deployment of advanced pose estimation techniques using MediaPipe and OpenCV to analyze human body postures from images. This involves a comprehensive process from data preprocessing to pose classification, aiming to develop a versatile tool applicable in various domains requiring human pose analysis.

2. Data and Methods

2.1. Information about the data (probably analysis of the data with some visualizations)

The MPII Human Pose Dataset comprises approximately 25,000 annotated images, each capturing a moment in human activity. These images exhibit a rich variety of poses, encompassing everyday actions like walking, sitting, and interacting with objects. Analyzing the MPII Human Pose Dataset offers profound insights into human motion and behavior. By examining the distribution of poses, we can uncover patterns and variations inherent in different activities. From simple standing postures to complex interactions, the dataset captures the full spectrum of human movement. This variability presents both challenges and opportunities for pose estimation algorithms, as models must handle diverse poses in real-world scenarios.



```python
import os
from matplotlib import pyplot as plt

image_file = "000071686.jpg"

image_path = os.path.join(root, image_file)
image = plt.imread(image_path)

plt.imshow(image)
plt.axis('off')
plt.show()
```

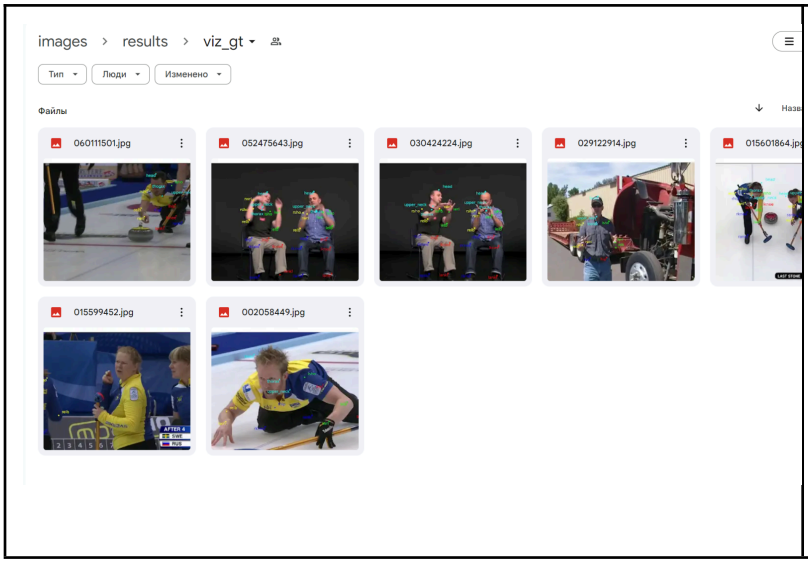
Link to official site: http://human-pose.mpi-inf.mpg.de/
Our Drive Link: https://drive.google.com/drive/folders/1ltDw2XSfj0FNuWJ_n9Fa5lKj-67NJWGU?usp=sharing

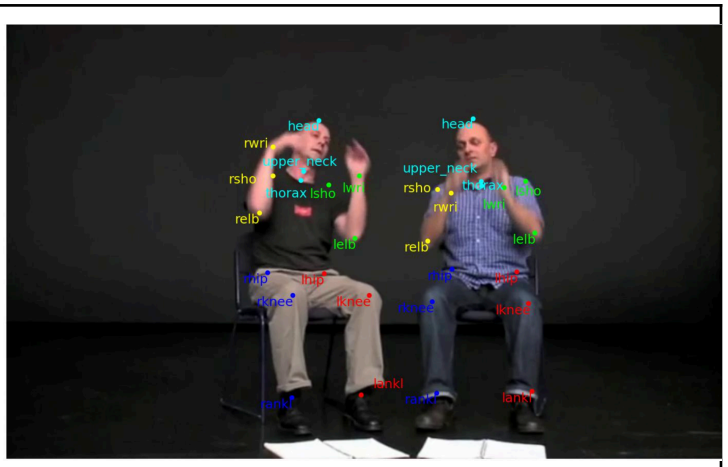
2.2 Description of the ML/DL models you used with some theory

Exploring models:

In exploring various models of pose estimation, we encountered two significant approaches:

MPII Human Pose Visualization: This method involves loading and visualizing the MPII Human Pose dataset using Python. The MPII Human Pose dataset provides a comprehensive collection of images annotated with the positions of various human body joints. By utilizing the joints.mat model, we were able to estimate the coordinates of key body parts such as shoulders, head, and legs. The process involves reading the dataset and interpreting the annotations to create visual representations of human poses. The sample output visualizations can be found in the results/viz_gt directory.

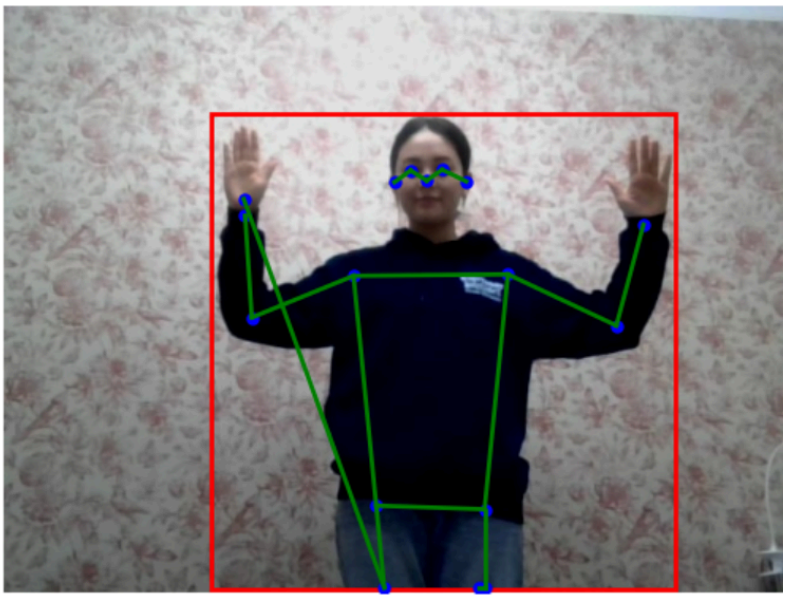


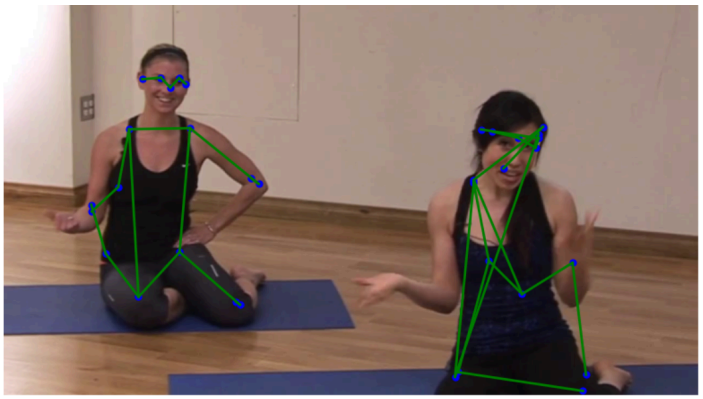


Usage of MPII Human Pose Visualization:

Link: https://github.com/meghshukla/MPII-Human-Pose-Visualization.git

Pytorch keypointrcnn_resnet50_fpn library: This approach utilizes the PyTorch library and specifically the keypointrcnn_resnet50_fpn model for pose estimation. This model employs deep learning techniques to detect and localize human keypoints using tensors. By leveraging the power of deep neural networks, this method can accurately identify the positions of key body joints in images or videos. The keypointrcnn_resnet50_fpn model is particularly notable for its robustness and accuracy in pose estimation tasks.





Usage of pytorch:

Link:
https://pytorch.org/vision/main/models/generated/torchvision.models.detection.keypointrcnn_resnet50_fpn.html

Final version model:

In our project we use the *MediaPipe* library, which is a popular open-source library developed by Google for building cross-platform machine learning solutions. Within MediaPipe, our script employs the Pose solution, a powerful feature designed for human pose estimation. Below is a detailed description of the underlying machine learning and deep learning models used in MediaPipe Pose, along with some relevant theoretical background:

MediaPipe Pose:

Framework: MediaPipe Pose utilizes a machine learning framework that applies convolutional neural networks (CNNs) to estimate the human body's pose from images. The underlying model identifies key points on the body, such as elbows, knees, shoulders, and hips, and infers the poses by predicting these points' locations in the image space.

Model Complexity Levels: MediaPipe Pose offers different model complexity levels, with your code specifically using model_complexity=2, which refers to a heavier model that typically yields more accurate results but requires more computational resources. This model complexity level influences the depth and structure of the underlying neural network.

Pose Landmarks: The core output of MediaPipe Pose is a set of landmarks, each corresponding to specific points on the human body. These landmarks are identified through a process that begins with feature extraction from the input image, followed by the application of a series of convolutional neural layers designed to progressively refine the prediction of landmark positions.

Heatmaps and Regression: Internally, the model generates heatmaps for each landmark, representing the probability of each pixel in the image being part of a particular keypoint. Additionally, it uses regression to directly predict the offset vectors from the keypoint positions to provide more precise localization.

BlazePose: The underlying model of MediaPipe Pose, particularly at higher complexity levels, is often referred to as BlazePose. BlazePose is designed to be efficient and fast, making it suitable for real-time applications on both mobile and desktop platforms. The model is trained on a large dataset of

images covering a wide variety of poses, backgrounds, and human subjects to ensure robustness and accuracy.

Applications: The accurate pose estimation provided by models like those in MediaPipe Pose enables a variety of applications, from augmented reality and fitness tracking to surveillance and human-computer interaction.

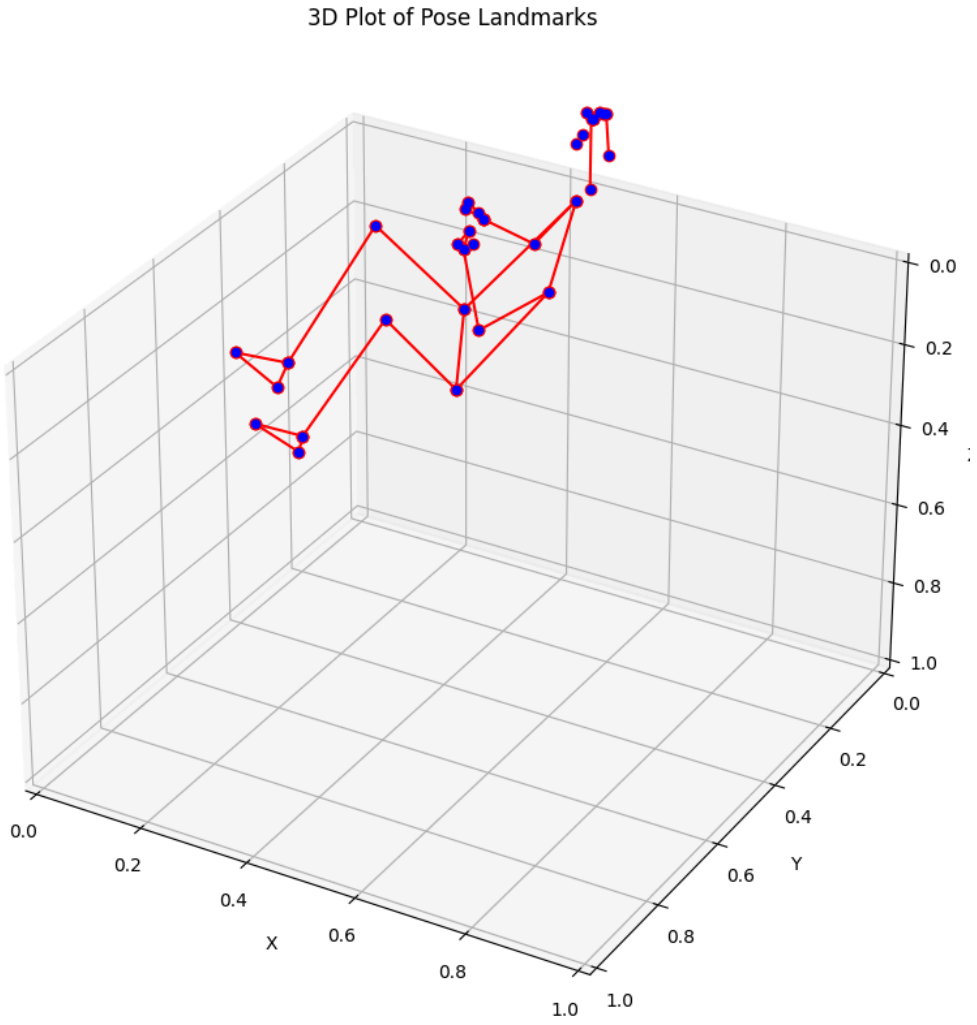
In our code, the MediaPipe Pose model provides a real-time, accurate estimation of human poses by detecting and tracking various body landmarks. The model's effectiveness stems from its deep learning foundation, particularly convolutional neural networks, which have revolutionized image-based tasks in computer vision. By adjusting parameters like model_complexity, you can balance between computational efficiency and pose estimation accuracy to suit your application's needs.

3. Results

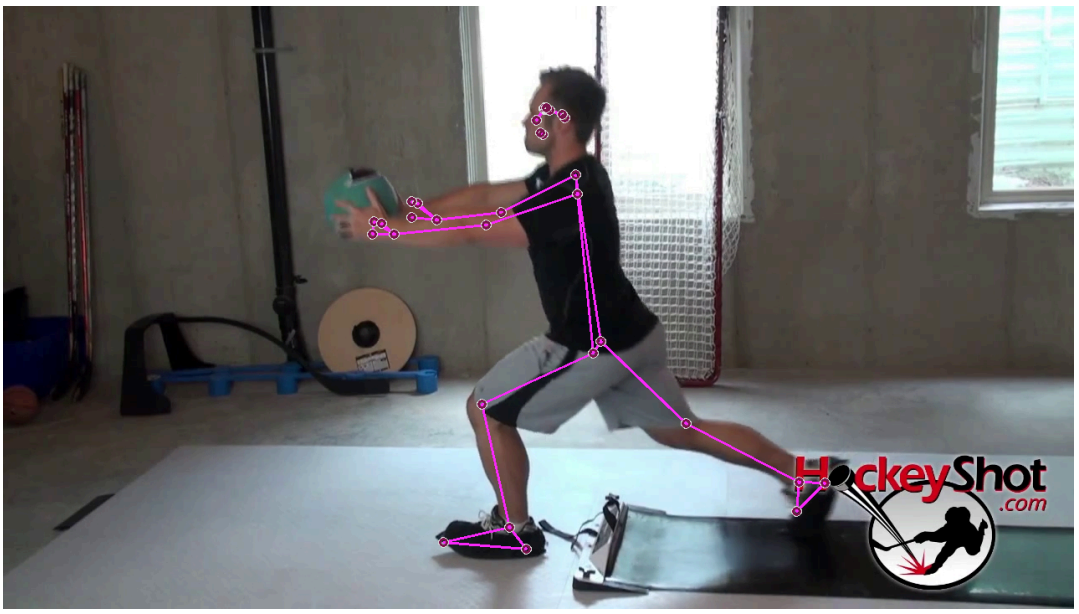
3.1. Results with tables, pictures and interesting numbers

In the first case we are applying MediaPipe Pose detection to identify human body landmarks and overlaying them onto the original image. The landmarks and connections between them are visualized as pink circles and purple lines, respectively, using MediaPipe's drawing utilities. This visualization is created by processing the image through the code, then displaying the annotated image with Matplotlib.

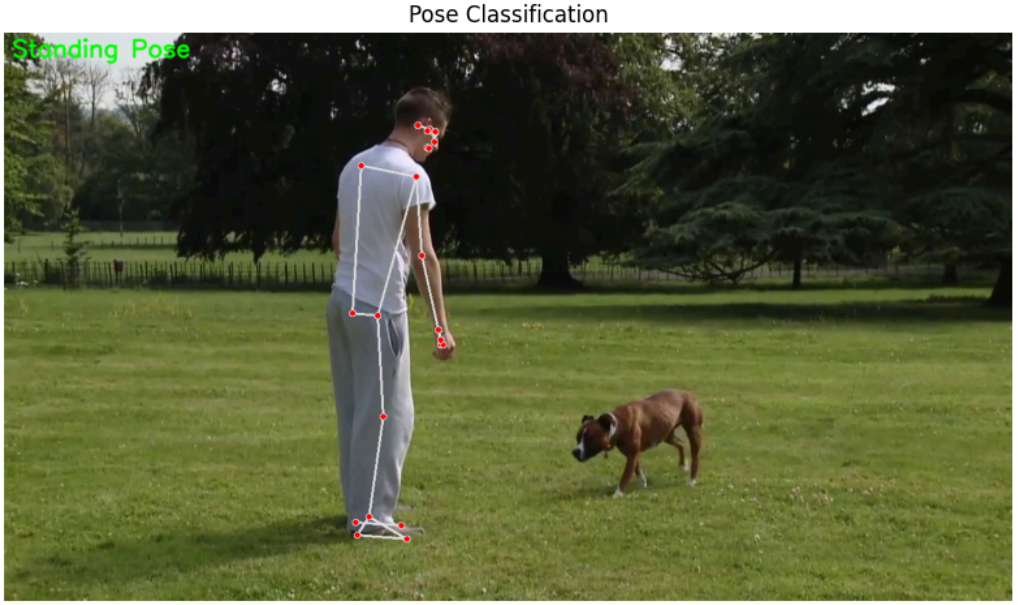
After that we visualize3D pose landmarks detected by MediaPipe. It creates a 3D plot, drawing connections between landmarks (skeleton) in red and marking individual landmarks as blue dots. The axes are set and inverted for a clearer view, showcasing the pose in a three-dimensional space. If no landmarks are detected, it outputs "No landmarks detected". And in the result we get the photo below:



Next step that we do is a function detect_and_plot_pose that reads an image from a given path, applies MediaPipe's pose detection to identify human pose landmarks, and then overlays these landmarks and their connections onto the original image. If pose landmarks are detected, the updated image is displayed with landmarks highlighted and connected by lines, demonstrating the human pose structure as in the provided photo. If no landmarks are detected, it informs the user. The example demonstrates pose detection in action, highlighting the practical application of MediaPipe's pose estimation technology in analyzing human postures.

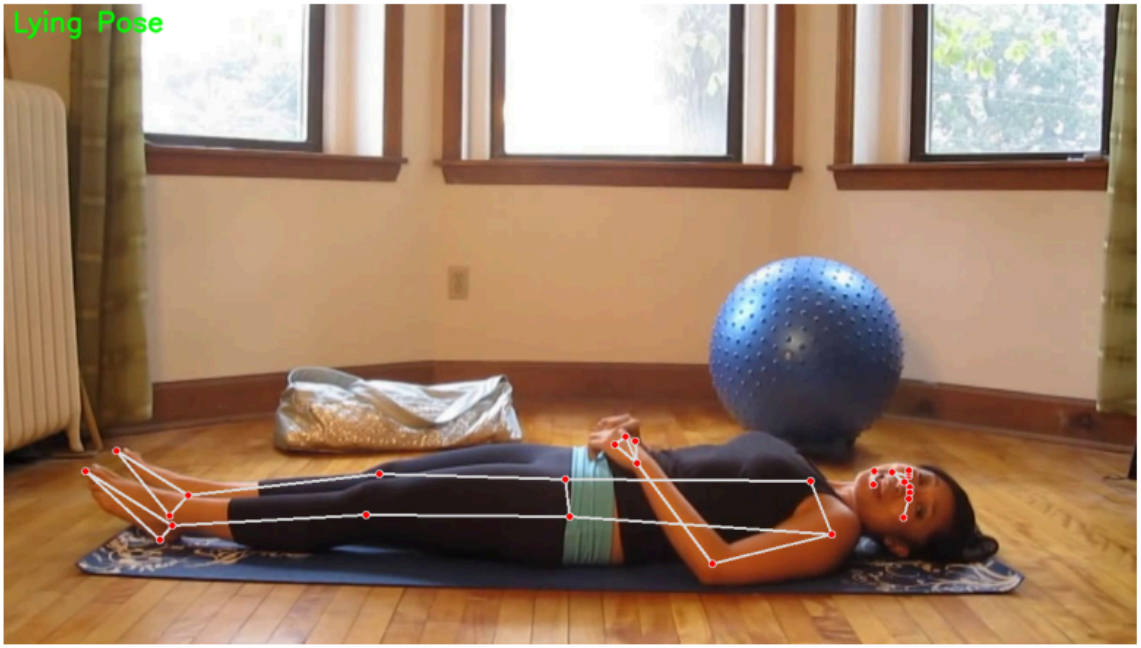


Finally to estimate the poses, get the name, keypoints(body parts) and connection between them we integrate MediaPipe and OpenCV for pose detection and classification in a static image. It employs MediaPipe's pose detection to find and annotate body landmarks, then calculates angles between specific landmarks to assess the posture. Using these angles, the classifyStandingPose, classifySittingPose, classifyLyingPose, classifyDownwardDogPose, classifyTPose function determine whether the individual is in a standing, sitting, lying, downward dog, t pose based on the alignment and straightness of the legs and the levelness of the shoulders and hips. The final classifications, such as "Standing Pose," or other pose are displayed on the image along with the detected pose landmarks, resulting in a visual output similar to the provided photos in the below:

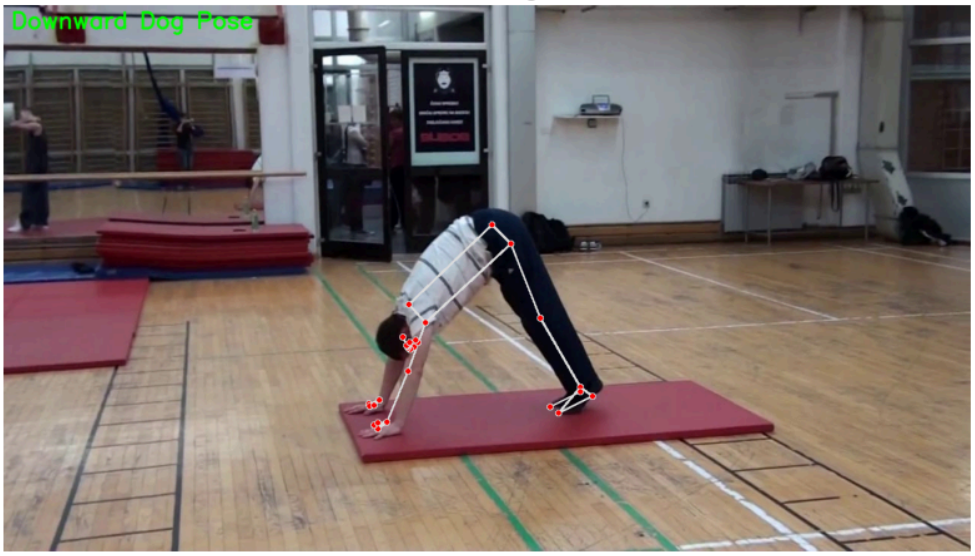


Sitting Pose

Pose Classification



Downward Dog Pose



T Pose

In the end, we employ a combination of Python libraries, including IPython, Google Colab, PIL, OpenCV, NumPy, and MediaPipe, to capture real-time photos and analyze body postures. And begin by capturing a photo through a JavaScript-based camera interface integrated within Google Colab notebooks, allowing real-time image capture directly in the notebook. This photo is then encoded and saved locally.

We've developed several functions to calculate the angles between these landmarks, which assist us in classifying the captured pose into categories like Standing, T Pose, Sitting, Downward Dog, or Lying, based on the relative positions and angles of the limbs and body. Each pose classification has specific criteria based on the angles and positions of the limbs.

Once a pose is detected and classified, our code annotates the image with the identified pose landmarks and labels it with the classified pose name. We display this annotated image with pose classification to provide immediate visual feedback. This process showcases our team's effort to integrate real-time pose classification into an accessible platform, enabling immediate and practical applications in fitness, health monitoring, or interactive learning environments. And you can see all the results that we record using photos in real time in our defense video, which we provided at the link.

4. Discussion

4.1. Critical review of results

This part critically examines the results of our project in several sections: accuracy of pose detection and classification, efficiency and user experience, and areas of criticism.

Pose detection and classification accuracy:

We chose to use MediaPipe due to its accuracy and efficiency in pose estimation. In the best and ideal scenarios, we still achieved excellent hit rates, but at the same time, we are fully aware that in real-world scenes, results can be affected by lighting, background complexity, and user diversity. The key is to keep our model and dataset up to date so that we can achieve greater accuracy in all cases that users may encounter.

Efficiency and performance:

The real-time performance of our system is very important when applying pose. We are very aware of how important it is to keep latency to a minimum and therefore have gone to great lengths when optimizing the code for Swift pose detection and classification. We understand that performance may vary

depending on devices and environment. As such, we are committed to further efficiency gains that will ensure our solution is available on a wider range of devices.

User Experience:

This makes the tool more convenient and easier to use. It has been shown that its setup may be too complex for a trained, casual, non-technical user to use the tool perfectly. The user interfaces include JavaScript and Google Colab, making the system more user-friendly. This includes a continued focus on user experience improvements and actionable feedback.

Stability and generalization:

We strive to make our system universally applicable, given the fact that it must work reliably in environments that are constantly changing. Thus, testing with different environments and users will become a very important part of our development process. Our goal is to ensure that our system is robust and can generalize well across different settings and populations.

Potential for improvement:

We are exploring ways to improve our project by further including more recognizable poses, ways to improve user feedback, and improve user interaction with the project to make it more accessible. Continuing to roll out our solution across even more platforms and devices will truly allow it to reach an even wider audience and thereby become even more effective.

Ethics and Privacy Considerations:

Together we agree on the highest importance of privacy and ethics associated with image processing. We are committed to maintaining the highest standards of data security and transparency in its use, with secure user consent in mind. Last but not least, eliminating possible bias and ensuring inclusivity in our pose detector is part of our ethics responsibilities.

In summary, the above project shows great potential in real-time pose detection and classification. However, we understand that much remains to be done. We believe that work in the above-mentioned area can not only strengthen the effectiveness of our solution, but also make it accessible and become an ethical standard. This team will always dedicate their time to develop our project based on user feedback and technological advancements to become a valuable and respected tool applicable in many fields.

4.2. Next steps

Moving forward, our team plans to enhance the project by increasing the diversity and size of our training dataset to improve pose detection accuracy across various user demographics and environments. These are things we will continue to work on to optimize our app for the best possible real-time performance on an even wider range of devices and continue to make our user experience even more accessible and engaging. We also want the library to be enriched by adding more detailed varieties of pose classifications and more detailed information on ethics and privacy issues. We will develop all improvements together with industry experts, respecting the user's opinions and preferences. Finally, the improved analysis was not limited to static images, but with the addition of video, it will facilitate dynamic motion assessment, thereby offering users complete feedback in real time.

Sources:

1. D. Pavllo, C. Feichtenhofer, D. Grangier and M. Auli, "3D Human Pose Estimation in Video With Temporal Convolutions and Semi-Supervised Training," EEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019 pp. 7745-7754. url: https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00794
2. Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2021. OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. IEEE Trans. Pattern Anal. Mach. Intell. 43, 1 (Jan. 2021), 172–186. https://doi.org/10.1109/TPAMI.2019.2929257
3. K. Sun, B. Xiao, D. Liu and J. Wang, "Deep High-Resolution Representation Learning for Human Pose Estimation," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019, pp. 5686-5696, doi: 10.1109/CVPR.2019.00584. url: https://openaccess.thecvf.com/content_CVPR_2019/papers/Sun_Deep_High-Resolution_Representation_Learning_for_Human_Pose_Estimation_CVPR_2019_paper.pdf
4. Xiao, B., Wu, H., Wei, Y. (2018). Simple Baselines for Human Pose Estimation and Tracking. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds) Computer Vision – ECCV 2018. ECCV 2018. Lecture Notes in Computer Science(), vol 11210. Springer, Cham. https://doi.org/10.1007/978-3-030-01231-1_29

https://openaccess.thecvf.com/content_ECCV_2018/papers/Bin_Xiao_Simple_Baselines_for_ECCV_2018_paper.pdf

5. Joo, H., Liu, H., Tan, L., Gui, L., Nabbe, B.C., Matthews, I., Kanade, T., Nobuhara, S., & Sheikh, Y. (2015). Panoptic Studio: A Massively Multiview System for Social Motion Capture. 2015 IEEE International Conference on Computer Vision (ICCV), 3334-3342. https://www.cs.cmu.edu/~hanbyulj/panoptic-studio/ICCV2015_SMC.pdf