DELHI TECHNICAL CAMPUS
GREATER NOIDA
Affiliated to GGSIPU and approved by AICTE & COA

A
REPORT

ON

Human Pose Estimation Using Machine Learning

Submitted in partial fulfillment of the requirement for the Lab (AIDS)

BACHELOR OF TECHNOLOGY (AIDS)

DELHI TECHNICAL CAMPUS GREATER NOIDA

SUBMITTED BY
Vanshika Shinde
(35118011923)
AIDS – 4th semester

SUBMITTED TO
Ms. Anjali Sardana
(HOD)

Affiliated to GGSIP University, New Delhi
Approved by AICTE & Council of Architecture

(SESSION 2024-2025)
DEPARTMENT OF ARTIFICIAL INTELLIGENCE
DELHI TECHNICAL CAMPUS, GREATER NOIDA
(Affiliated by Guru Gobind Singh Indraprastha University, New Delhi)

ACKNOWLEDGEMENT

This thesis is a reflection of the unwavering support and boundless love I received from my family, Delhi Technical Campus, and the project teammates during this challenging academic pursuit. I owe an immense debt of gratitude to my parents, who nurtured my curiosity and supported my educational endeavors from the very beginning.

I sincerely thank the technical staff (Ms. Anjali Sardana) whose expertise in managing lectures and projects was crucial for my experience. Her patience and readiness to assist at all times have left a profound impact on the completion of my project.

My appreciation also goes to the opportunity provided by DTC, whose resources and assistance have been invaluable. I would also like to acknowledge my peers for their camaraderie and the stimulating discussions that inspired me throughout my academic journey. Their collective wisdom and encouragement have been a cornerstone of my learning experience.

ABSTRACT

BRIEF SUMMARY

Human Pose Estimation (HPE) Using Machine Learning is a cutting-edge computer vision task focused on detecting and analyzing the positions of the human body from images or video data.

PROBLEM STATEMENT

In today's rapidly advancing digital age, understanding human movement and posture is critical across various domains, such as healthcare, sports, human-computer interaction, and entertainment

OBJECTIVE

To design and implement an innovative and user-friendly Human Pose Estimation application that leverages cutting-edge technologies like Python, OpenCV, Mediapipe, and Streamlit. This project aspires to make complex motion tracking and analysis accessible to everyone, fostering applications in fitness, sports, gesture recognition, and beyond, while maintaining exceptional accuracy, speed, and interactivity.

METHODOLOGY

Frontend features

- Interactive GUI: A simple yet powerful Streamlit-based interface to handle all inputs:
- File Upload: Upload images or videos

Real-Time Webcam Feed

- Detect poses dynamically with a live webcam feed
- Instant results

Backend Workflow

- Image Mode
- Webcam Mode

KEY RESULTS

🎬 **Accurate Pose Detection**
🎬 **Real-Time Performance**
🎬 **User-Friendly Interface**
🎬 **Multi-Input Compatibility**
🎬 **Robust Backend Workflow**
🎬 **Wide Application Potential**
🎬 **Scalability**

CONCLUSION

The Human Pose Estimation application successfully combines advanced computer vision with user-friendly design, delivering accurate, real-time pose detection from images, videos, and live webcam feeds. Built with Python, OpenCV, Mediapipe, and Streamlit, it provides a versatile platform for applications in fitness, sports, and gesture recognition. Its modular design ensures scalability, making it adaptable to future enhancements like 3D pose estimation. This project highlights the potential of technology to simplify complex tasks and make cutting-edge tools accessible to all.

TABLE OF CONTENT

LIST OF FIGURES

Introduction

Problem Statement:

Understanding human movements and body postures is a complex task, particularly in fields like sports, healthcare, and surveillance. Without automated systems like Human Pose Estimation, activities such as motion analysis and injury prevention rely on manual methods, which are often slow, labor-intensive, and prone to errors. This creates a need for efficient and accurate solutions that can address these challenges effectively. This project is significant because it addresses key challenges and opportunities in understanding human movement and posture through accessible and innovative technology.

Objective:

Objectives of the Human Pose Estimation Project are:

□ **Develop an Accurate Pose Estimation System:**

Implement a robust machine learning application to accurately detect and analyze human body key points from images, videos, and live webcam feeds.

□ **Enhance Accessibility:**

Create a user-friendly platform using Python, OpenCV, Mediapipe, and Streamlit to make pose estimation technology accessible to a broader audience, even non-technical users.

□ **Enable Real-Time Processing:**

Ensure the system provides real-time feedback for live applications, such as fitness tracking, motion analysis, and gesture recognition.

□ **Support Multiple Input Modes:**

Design the application to seamlessly handle static images, video files, and live webcam feeds, catering to various use cases.

□ **Promote Versatility and Scalability:**

Build a modular and scalable system that can be adapted for future advancements, such as 3D pose estimation or activity recognition.

□ **Foster Practical Applications:**

Enable the system to be used in diverse fields like healthcare, sports, gaming, and human-computer interaction, contributing to innovation and problem-solving in these areas.

□ **Democratize Advanced Technology:**

Make advanced pose estimation tools affordable and accessible, reducing dependency on expensive equipment and specialized software.

Scope of the Project:

This project's scope extends from being an accessible pose estimation tool to serving as a foundation for innovative applications across various industries.

Core Features

- Develop a robust application to detect human body key points and estimate poses from static images, videos, and live webcam feeds.
- Provide a visual overlay of detected pose landmarks for better understanding and analysis.

Technology Utilization

- Leverage Python, OpenCV, Mediapipe, and Streamlit to ensure efficient processing, user-friendly interfaces, and scalability.
- Employ machine learning techniques for high accuracy and real-time performance.

Supported Input Formats

- **Image Mode:** Handle popular image formats (e.g., JPG, PNG).
- **Webcam Mode:** Enable live pose detection through real-time webcam feeds.

Scalability

- Design the system to adapt to advanced capabilities like 3D pose estimation.
- Ensure modular architecture to integrate additional features or technologies seamlessly.

Accessibility Goals

- Make pose estimation technology affordable and intuitive for non-technical users.

- Provide solutions accessible to professionals, hobbyists, and developers alike.

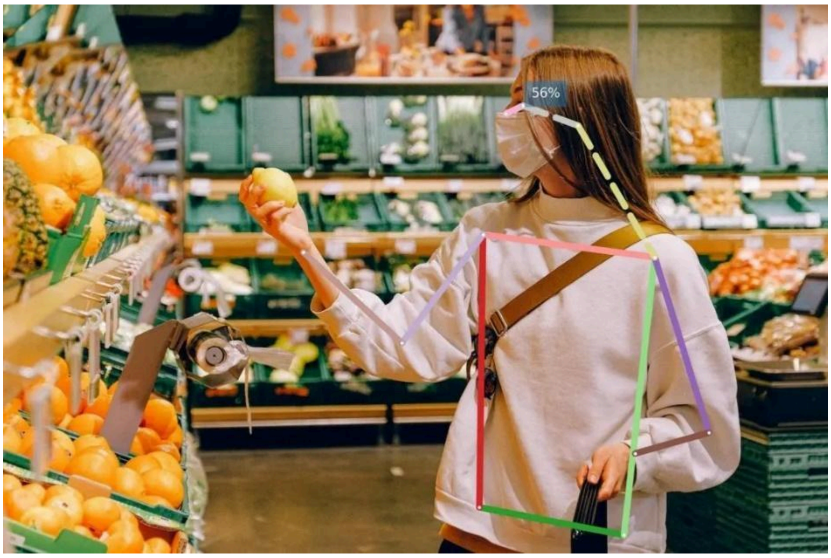


Figure 1: Reference to model

Literature Survey

Literature Review

Human pose estimation has gained significant attention in recent years due to advancements in deep learning and computer vision. Traditional approaches relied on handcrafted features and shallow machine learning models, which struggled with complex poses, occlusions, and variability in environments. The introduction of deep convolutional neural networks (CNNs) revolutionized the field, enabling more accurate keypoint detection and pose estimation. Notable early models include OpenPose, which introduced a multi-stage architecture for detecting keypoints and associating them with individuals in multi-person scenarios. Similarly, DeepPose, developed by Google, leveraged CNNs for direct regression of joint locations, marking a shift from traditional graphical models. These methods laid the groundwork for further developments by highlighting the importance of robust feature extraction and data-driven learning.

Recent work has focused on improving efficiency, accuracy, and scalability. Techniques such as heatmap-based keypoint detection, as seen in the High-Resolution Network (HRNet), have enhanced precision by preserving spatial information. Lightweight architectures like MobilePose and BlazePose have been developed for real-time applications on mobile and embedded devices. Transformer-based models, like PoseFormer, have recently shown promise in capturing long-range dependencies in pose estimation tasks. Despite these advancements, challenges such as occlusion handling, domain generalization, and ethical considerations remain active areas of research. Efforts are also directed toward creating larger, more diverse datasets like COCO and MPII to improve model robustness and ensure fairness in deployment across various real-world scenarios.

.

Some Existing Models, Techniques, or Methodologies

Traditional Methods:

- Pictorial Structures (Felzenszwalb et al., 2005): Used probabilistic graphical models to represent the human body as a set of interconnected parts.
- Deformable Part Models (DPM): Combined part-based representations with handcrafted features like HOG (Histogram of Oriented Gradients) for detecting and localizing body parts.

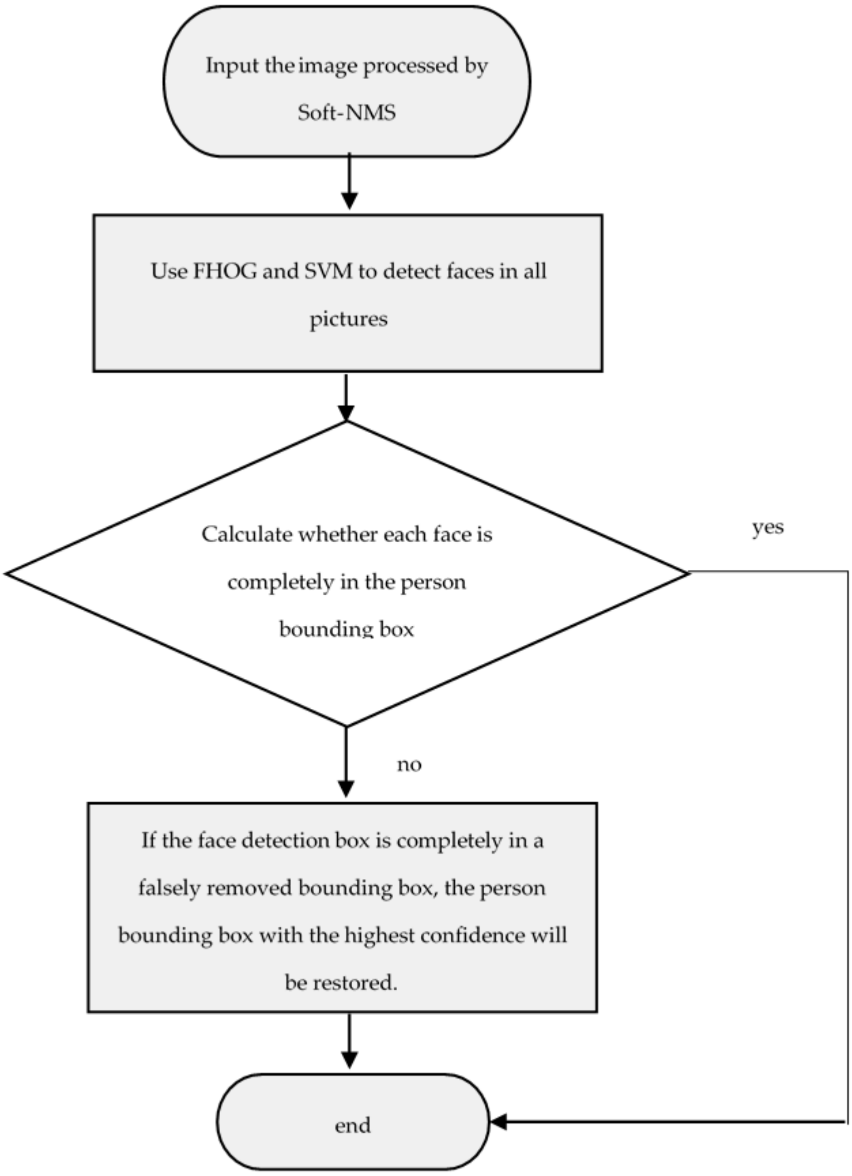

Input the image processed by Soft-NMS
Use FHOG and SVM to detect faces in all pictures
Calculate whether each face is completely in the person bounding box
yes
no
If the face detection box is completely in a falsely removed bounding box, the person bounding box with the highest confidence will be restored.
end

Figure 2: Face Detection and Bounding Box Refinement Workflow

Deep Learning-Based Models:

- DeepPose (Toshev & Szegedy, 2014): The first CNN-based approach to regress body keypoints, marking the shift to deep learning.
- Hourglass Networks (Newell et al., 2016): Introduced a stacked architecture to refine pose predictions across multiple stages.
- OpenPose (Cao et al., 2017): A bottom-up method that detects body parts and links them into poses using part affinity fields (PAFs).
- Mask R-CNN (He et al., 2017): A top-down approach that extends Faster R-CNN for instance segmentation and pose estimation.

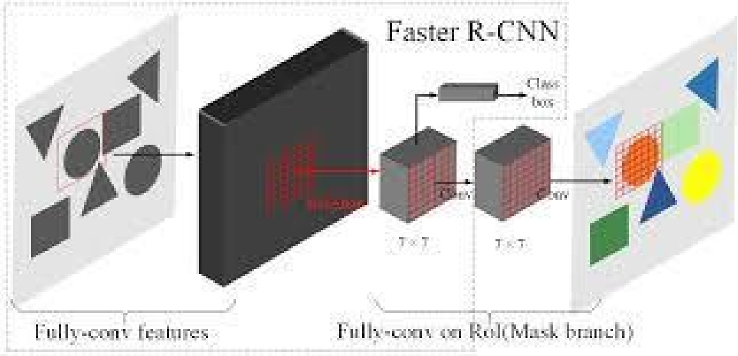


Figure 3: Faster R-CNN Architecture for Object Detection and Segmentation

Transformer-Based Methods:

- PoseFormer (Zheng et al., 2021): Used transformer architectures to model long-range dependencies between joints, improving pose estimation for complex and occluded poses.

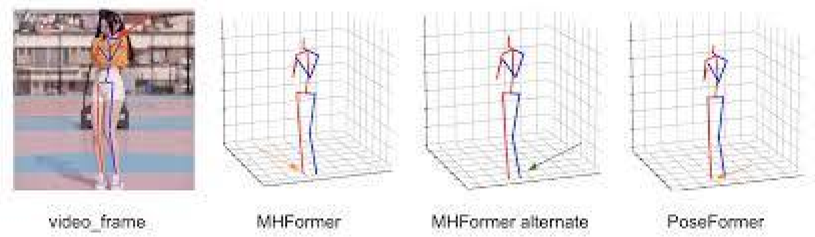


Figure 4: Comparison of 3D Pose Estimation Models: MHFormer, MHFormer Alternate, and PoseFormer

Temporal Models for Video Pose Estimation:

- PoseLSTM (Song et al., 2017): Leveraged recurrent neural networks to capture temporal dynamics of body movement in videos.
- ST-GCN (Yan et al., 2018): Introduced spatio-temporal graph convolutional networks for video-based pose estimation, modeling both spatial and temporal dependencies.

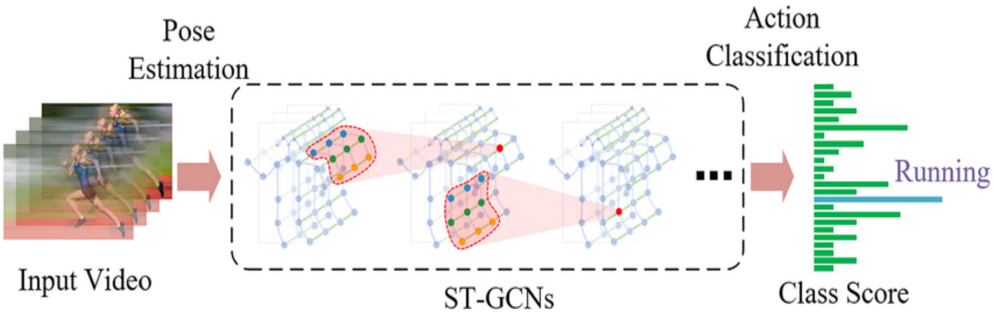


Figure 5: Visual Comparison of Pose Estimation Techniques Across Models

Multi-Person Pose Estimation Techniques:

- Top-Down Methods: Detect individuals first (e.g., using Faster R-CNN) and then estimate their poses (e.g., Mask R-CNN).
- Bottom-Up Methods: Detect all keypoints first and group them into poses (e.g., OpenPose).
- Hybrid Methods: Combine elements of both approaches to improve robustness in crowded scenes.

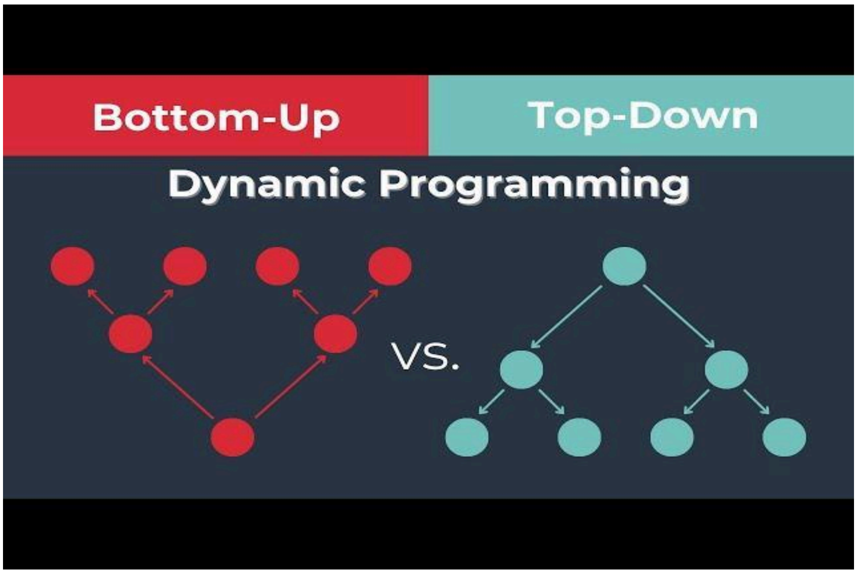


Figure 6: Dynamic programming

Limitations of Human Pose Estimation Using Machine Learning

1. **Data Issues**: Limited or imbalanced datasets and biased training data reduce model generalization.
2. **Occlusion**: Difficulty detecting body parts hidden by objects or other people.
3. **Real-Time Performance**: High computational requirements hinder real-time applications, especially on resource-constrained devices.
4. **Environmental Factors**: Poor lighting, cluttered backgrounds, weather conditions, and fast movements degrade accuracy.
5. **Model Accuracy**: Challenges in keypoint localization, joint misalignment, and extreme poses.
6. **Scalability**: High resource demands for large-scale deployment or adaptation to new domains.
7. **Privacy & Ethics**: Privacy concerns, biased results, and lack of interpretability.

Motivation:

Machine learning (ML) is a transformative technology that empowers systems to learn from data and improve their performance without being explicitly programmed. The motivation to adopt ML lies in its ability to address complex challenges and unlock new possibilities across industries.
The motivation stems from potential applications in healthcare and rehabilitation, sports and fitness, human-computer interaction, gaming and animation, the entertainment industry, security and surveillance, education and training, retail and shopping, and robotics and automation. This project demonstrates immense potential across diverse fields, showcasing the value of technology in simplifying and enriching human activities.

The Human Pose Estimation project revolutionizes motion analysis by providing affordable, user-friendly impacts for industries like healthcare, fitness, sports, and entertainment. It enhances well-being through real-time posture correction and injury prevention while fostering innovation in gaming, VR, and accessibility systems. With its dynamic, scalable design, the project promotes inclusivity, supports education, and drives economic growth by making advanced technology widely accessible.

System Design and Methodology

OVERVIEW

The image describes the system design and workflow for a human pose estimation project using machine learning. Here's a detailed explanation of the system design, its components, and the working mechanism:

The system supports two input modes:

1. Image Input Mode
2. Webcam Input Mode

These inputs are processed using MediaPipe Pose, a machine-learning-based solution that detects human body landmarks from images or video streams.

ARCHITECTURE/ DIAGRAM

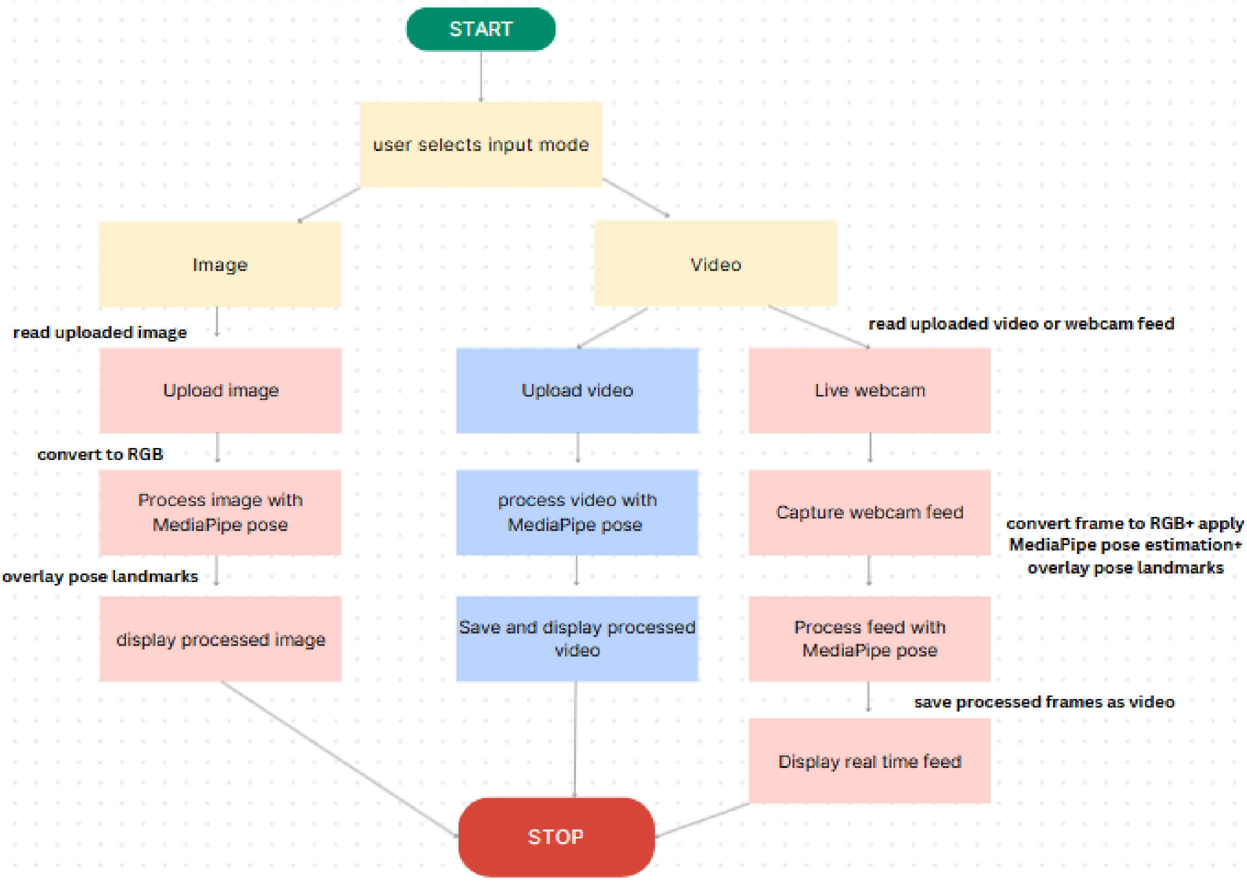

START
user selects input mode
Image
Video
read uploaded image
read uploaded video or webcam feed
Upload image
Upload video
Live webcam
convert to RGB
Process image with MediaPipe pose
process video with MediaPipe pose
Capture webcam feed
convert frame to RGB+ apply MediaPipe pose estimation+ overlay pose landmarks
overlay pose landmarks
display processed image
Save and display processed video
Process feed with MediaPipe pose
save processed frames as video
Display real time feed
STOP

Figure 7: flowchart for the project

TOOLS AND TECHNOLOGIES USED

- **Deployment Platforms**:
  - **Local Applications**:

Deployed as a desktop application (using Python and OpenCV).

- o **Web Applications**:

Interactive GUI: A simple yet powerful Streamlit-based interface to handle all inputs: images, videos, and live webcam feeds.

File Upload: Upload images or videos directly in popular formats like JPG, PNG, MP4, AVI, etc.

Real-Time Webcam Feed: Detect poses dynamically with a live webcam feed

.

- **Steps in Deployment**:
  - o **Integrate MediaPipe Pose**: Use the pre-trained MediaPipe Pose model, which is highly optimized for real-time landmark detection.
  - o **Optimize Performance**:
    - ▪ Use techniques like quantization and pruning to reduce latency.
    - ▪ Ensure GPU or TPU support for real-time processing.
  - o **Build User Interface (UI)**:
    - ▪ Design UI for easy input selection (image/video).
    - ▪ Include options to upload media or use live webcam feeds.

WORKING

The system processes either static images or video streams in real time. The workflow is as follows:

- **Image Mode**

1. **Upload Image**:
   - o User uploads a static image containing a person.
2. **Convert to RGB**:
   - o Image is pre-processed and converted to RGB format.
3. **Process with MediaPipe Pose**:
   - o MediaPipe detects human body landmarks in the image.

4. **Overlay Pose Landmarks**:
   - The system visualizes detected landmarks and pose connections on the image.
5. **Display Processed Image**:
   - The processed image is displayed back to the user.

- **Live Webcam Feed**:
  1. Real-time video feed is captured from the webcam.
  2. Each frame is processed with MediaPipe Pose for landmark detection.
  3. The output is displayed in real-time with pose landmarks.

Applications

Human pose estimation has a wide range of real-world applications:

- **Fitness and Sports**:
  - Track exercise form and posture in real-time fitness apps.
  - Analyze athletes' movements for performance enhancement.
- **Healthcare and Rehabilitation**:
  - Monitor recovery exercises for patients in rehabilitation.
  - Detect improper body posture to prevent musculoskeletal disorders.
- **Gaming and Entertainment**:
  - Enable motion-based gaming experiences.
  - Create engaging AR/VR experiences.
- **Surveillance and Security**:
  - Monitor human activities and detect anomalies in surveillance systems.
- **Education**:
  - Teach body movements and dance routines interactively.
- **Human-Computer Interaction (HCI)**:
  - Enable gesture-based control of devices.

FLOWCHART AND DIAGRAM

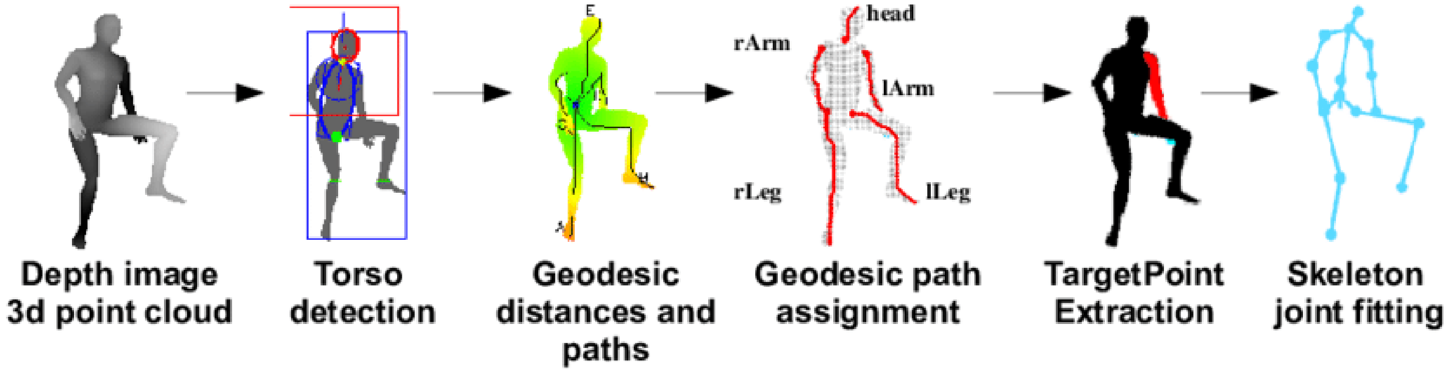


Figure 8: Proposed Approach for Human Pose Estimation

IMPLEMENTATION

MODULE WISE DESCRIPTION:

Requirement Specification

Functional Requirements

These define the specific functionalities and operations the system must perform.

Input Handling:

- Image Input
    - o The system should allow users to upload static images for processing.
- Video Input
    - o The system should allow users to upload video files or access live webcam feeds.

Preprocessing:

- Convert input images and video frames to the RGB format for compatibility with MediaPipe Pose.

Real-Time Processing:

- Ensure real-time performance when processing live webcam feeds, with minimal latency.

Error Handling:

- Provide error messages for invalid input formats or unsupported file types.
- Handle cases where human poses cannot be detected (e.g., poor lighting, obstructions).

Device Compatibility:

- The system should work on various devices, including desktops, laptops, and mobile platforms if deployed as a web or mobile application.

Non-Functional Requirements

These define the quality attributes and constraints of the system.

SCALABILITY:

- Handle multiple people in a single image or video frame.
- Support higher-resolution inputs without significant performance degradation.

ACCURACY:

- Achieve high accuracy in detecting human pose landmarks, even in diverse backgrounds and lighting conditions.

RELIABILITY:

- The system should function consistently across different environments and handle various user inputs without crashing.

USABILITY:

- Provide an intuitive interface for non-technical users to operate the system easily.

PORTABILITY:

- The system should be portable across platforms:
  - Use MediaPipe Pose for cross-platform support (e.g., Python, Android, iOS).

SECURITY:

- Ensure that user-uploaded media is handled securely and not stored unnecessarily.

EXTENSIBILITY:

- Allow future integration with other pose estimation models or additional features, such as gesture recognition or activity classification.

RESOURCE EFFICIENCY:

- Optimize resource usage to run efficiently on devices with limited hardware capabilities (e.g., mobile devices or edge devices)
- Mention the tools and technologies required to implement the solution.

Hardware Requirements:

1. Programming language: Python
2. Libraries:

Streamlit: Used for building web applications

MediaPipe: Pre-trained library specifically for pose estimation.

OpenCV: Used for image and video input handling, preprocessing, and visualization.

Software Requirements:

1. Processor: Intel i7, AMD Ryzen 7 CPU
2. RAM:

CPU-Based: 8–16 GB

GPU-Based: 4 GB (GPU VRAM) + 8 GB (System RAM)

3. Storage: minimum 6 GB
4. Internet Connection: Required for Streamlit application, deployment and testing.

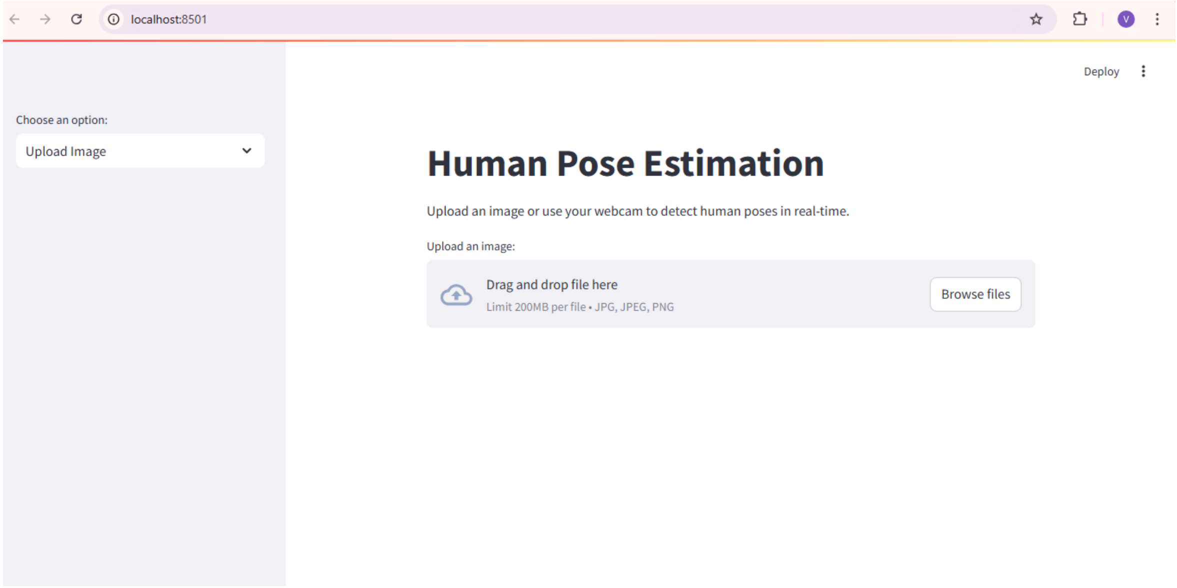
KEY FUNCTIONALITIES

This pose estimation system incorporates several user-centric features:

- **Dual Input Modes**:
  - Supports static images and dynamic video streams.
- **Real-Time Processing**:
  - Processes webcam feeds and displays results instantly.
- **High Accuracy**:
  - Leverages MediaPipe's pre-trained models for accurate pose detection.
- **Visualization**:
  - Overlays pose landmarks and skeletons on the input media.
- **Customizability**:
  - Can be tailored for specific applications (e.g., fitness, gaming).
- **Cross-Platform Support**:
  - Deployable on desktops, mobile devices, or web browsers.
- **User-Friendly Interface**:
  - Easy-to-use system with options for uploading or capturing input.
- **Scalability**:

- Optimized for processing multiple individuals in an image or video.

USER INTERFACE SNAP SHOTS



Human Pose Estimation

Upload an image or use your webcam to detect human poses in real-time.

Upload an image:

Figure 9: Streamlit web interface

This page is the Streamlit Web Interface for the Human Pose Estimation project. Here's what each element represents:

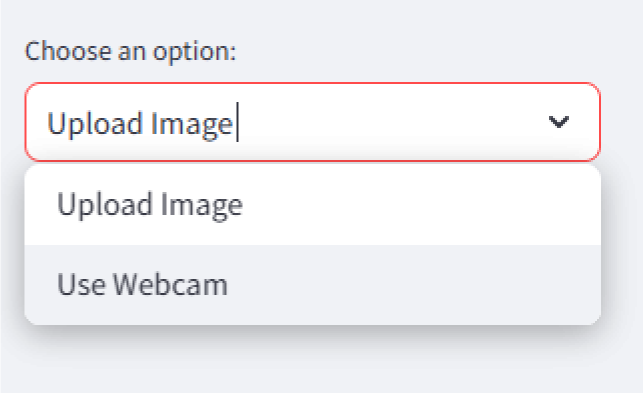
Key Features:

1. **Title & Description**
   - Displays the app's purpose: detecting human poses in uploaded images or via webcam.
2. **Sidebar Dropdown**
   - Options:
     - **Upload Image**: For analyzing images.
     - **Use Webcam**: For real-time pose detection (not selected here).

3. **Image Upload Section**
    - Allows users to drag/drop or browse image files (JPG, JPEG, PNG).

How It Works

- Upload an image, and the app processes it using **MediaPipe Pose**.
- Key body landmarks are detected, overlaid, and displayed directly in the browser.



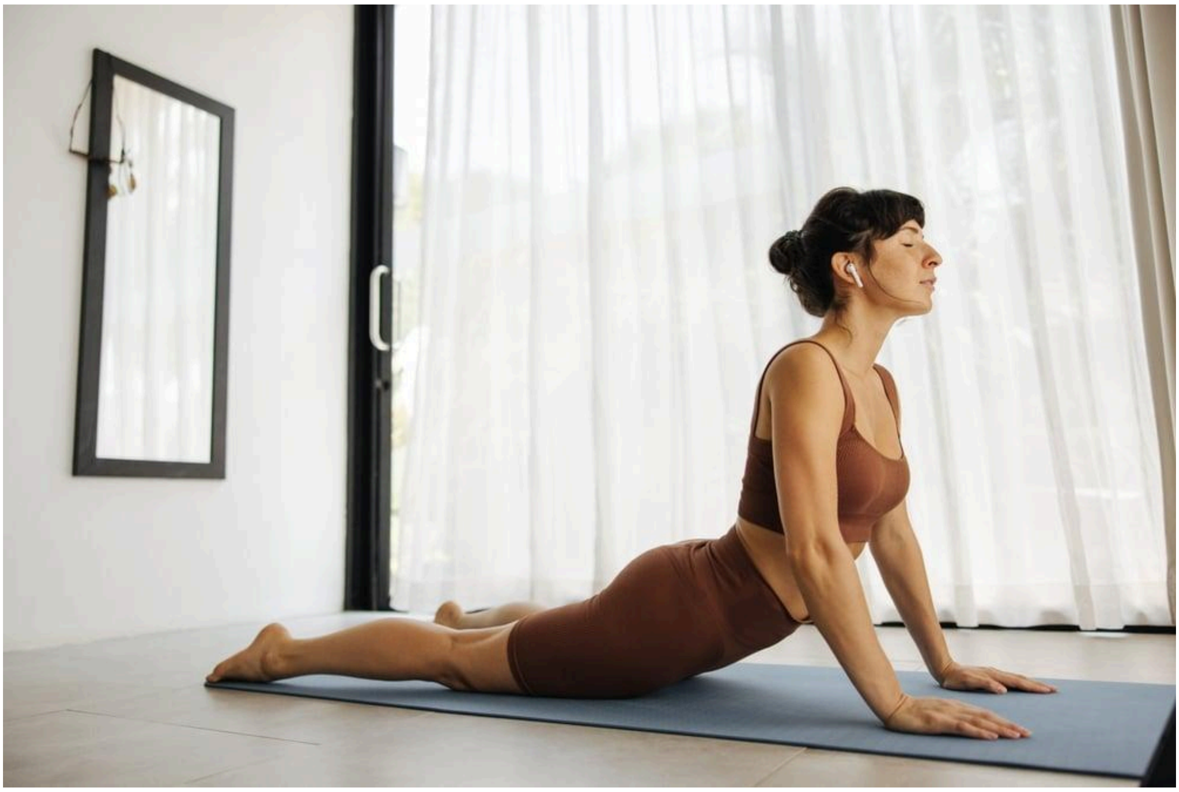
Choose an option:

Upload Image

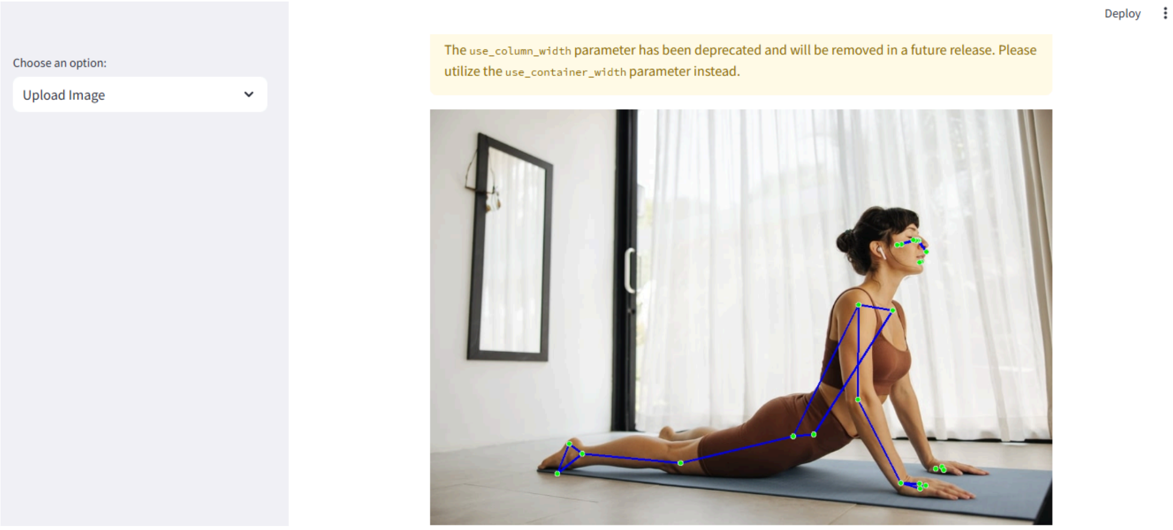
Figure 10: Sidebar dropdown for inputs

RESULTS

INPUT IMAGE1:



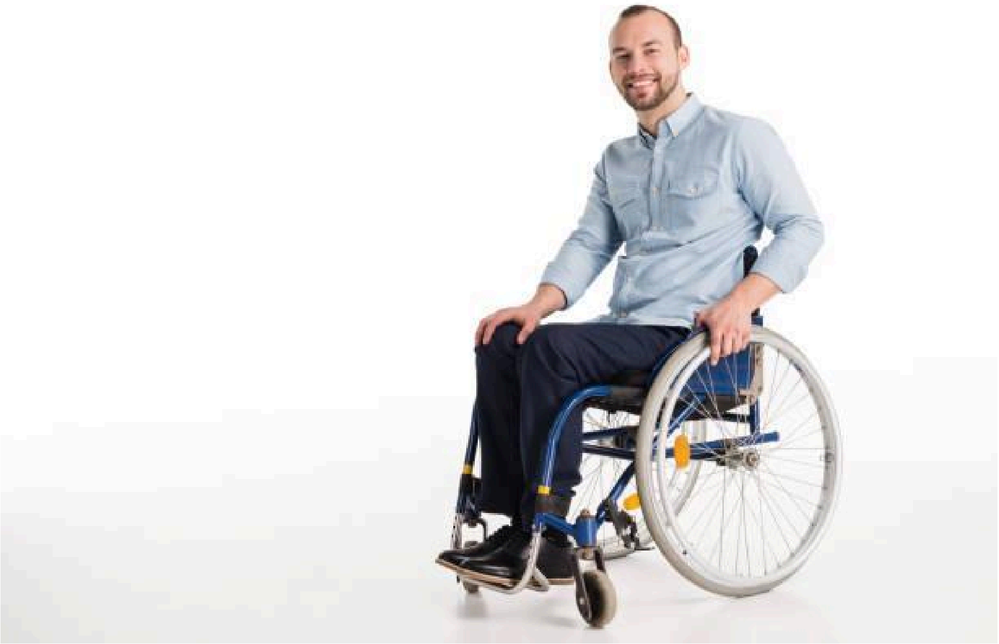
OUTPUT IMAGE1:



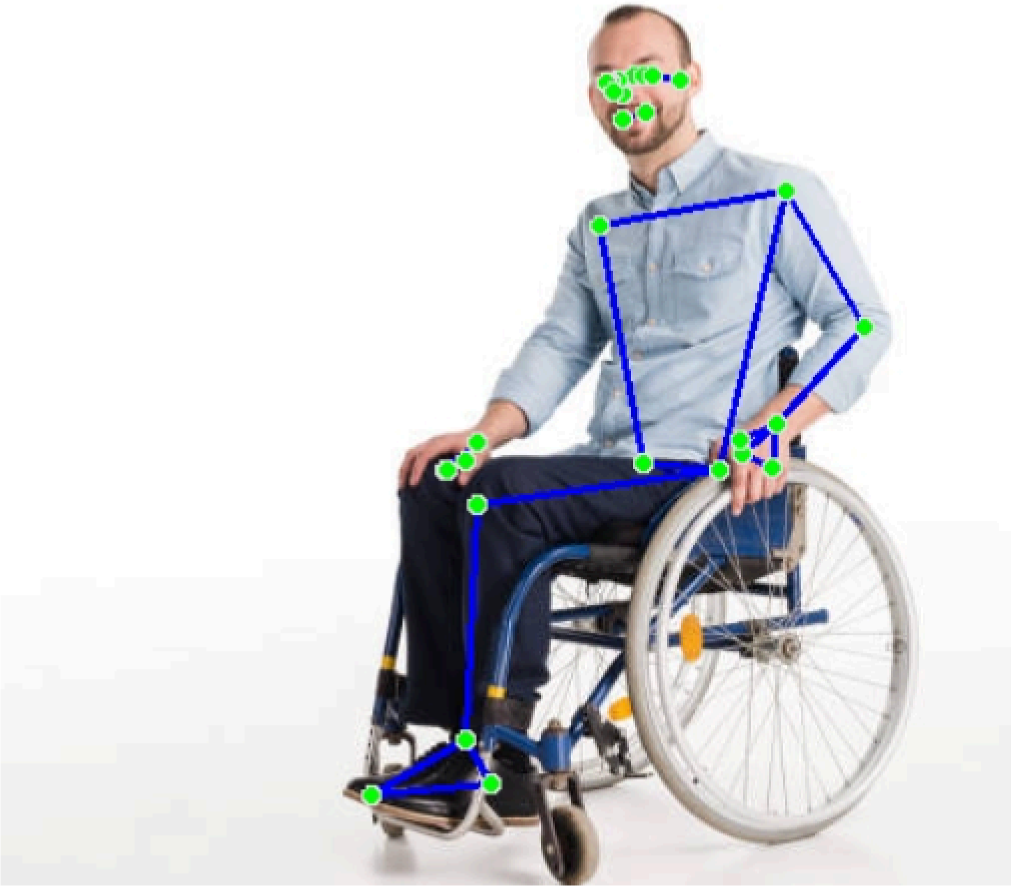
Choose an option:
Upload Image

The `use_column_width` parameter has been deprecated and will be removed in a future release. Please utilize the `use_container_width` parameter instead.

INPUT IMAGE2:



OUTPUT IMAGE2:



PERFORMANCE ANALYSIS FOR HUMAN POSE ESTIMATION (OPENCV + STREAMLIT)

1. Accuracy & Keypoint Detection

- **PCK (if available):** If you're using a model like OpenPose or MediaPipe via OpenCV, check how well keypoints are placed compared to ground truth (if you have labeled data).
- **Observation-Based Evaluation:** Since exact PCK/mAP may not be available without labeled data, you can:
  - Visually inspect overlayed keypoints on video.
  - Log false detections/missing joints.
  - Use a **confidence threshold** to filter weak keypoints.

2. Real-Time Performance

- **Frame Rate (FPS):**
  - Use time.time() around your frame loop to calculate average FPS.
  - Aim for **15–30 FPS** for smooth real-time experience.
- **Latency:** Streamlit adds a bit of overhead — keep model inference and rendering optimized.

3. Robustness Testing

- **Test Cases:**
  - Different lighting conditions.
  - Varying body sizes and angles.
  - Occlusion (e.g., arms behind body).
- **Observed Issues:** (customize this for your case)
  - ❌ Sometimes misses joints in low light.

  - ✅ Consistent on front-facing poses.
  - ❌ Lag during multi-person scenes.

4. Visual Quality in Streamlit

- **UI Fluidity:**
  - Use st.image() efficiently with channels="BGR" to reduce redraw lag.
  - Cache non-changing components (like loading the model).
- **Overlay Clarity:**
  - Check keypoint visibility (color, size, connections).
  - Allow toggling between original frame and pose overlay.

GitHub Link for Code:

https://github.com/vanshika847/human-pose-estimation-using-machine-learning

Conclusion

The project bridges the gap between cutting-edge machine learning technology and practical real-world applications, significantly enhancing automation, human interaction, and quality of life across multiple domains. It lays the groundwork for future innovations in human motion understanding while driving forward research and industry adoption.

LIMITATIONS

1. Accuracy Limitations

- Struggles with complex poses, fast movements, or occluded joints.
- May not perform well on non-frontal or side-angle body orientations.
- Lighting conditions significantly affect detection quality.

2. Model Limitations

- If using lightweight models (e.g., MediaPipe or OpenPose via CPU), they may sacrifice precision for speed.

- Pre-trained models may not generalize well to custom environments or unusual body types.

3. Performance Constraints

- Real-time speed can drop on lower-end CPUs (Streamlit + OpenCV rendering overhead).
- Multi-person detection may not be supported or could lead to heavy lag.
- Pose estimation models can be computationally intensive, especially without GPU acceleration.

4. Streamlit Limitations

- Not ideal for high-speed streaming — image updates may lag slightly depending on network/load.
- Lacks built-in support for video controls, pausing, or frame-by-frame review.

5. No Posture Classification / Feedback

- While keypoints are detected, there's no built-in analysis of posture (e.g., form correctness, pose similarity).
- System doesn't flag bad posture or offer corrections — just visualizes pose.

6. No Built-in Evaluation Metrics

- Lacks automatic computation of metrics like PCK, mAP, or OKS, unless you implement ground-truth comparison manually.
- Relies heavily on visual inspection for performance validation.

FUTURE ENHANCEMENTS:

Suggestions for improving the model or addressing any unresolved issues in future work:

1. Increase Model Accuracy and Robustness
   - Incorporate Advanced Architectures: Use advanced deep learning architectures like HRNet, Transformer-based pose models, or EfficientPose for better performance.
   - Train with Larger Datasets: Incorporate more diverse and larger datasets like COCO, Human3.6M, or PoseTrack to enhance the model's ability to generalize across different poses, environments, and occlusions.
   - Data Augmentation: Apply advanced augmentation techniques (e.g., rotation, flipping, occlusion, and noise addition) to make the model more robust to real-world variations.

2. Optimize for Real-Time Applications
   - Model Compression: Use techniques like pruning, quantization, and knowledge distillation to reduce the model size while maintaining accuracy.
   - Edge Deployment: Optimize the model for edge devices using frameworks like TensorFlow Lite, ONNX Runtime, or MediaPipe to improve inference speed and power efficiency.

3. Improve Occlusion Handling
   - Multi-Camera Inputs: Use data from multiple camera angles to improve pose estimation in occlusion cases.
   - Temporal Modeling: Integrate temporal models (e.g., LSTMs, GRUs, or Transformers) to predict poses in videos with occluded frames based on context from previous frames.

4. Support for Complex Poses
   - Train the model specifically for complex and unusual poses, such as yoga, gymnastics, or dance, by including datasets or custom-labeled images/videos tailored to these activities.

5. **Advanced Real-Time Features**
   - **Motion Prediction**: Use predictive modeling to estimate the next movement based on current motion trends, improving applications like autonomous surveillance or gesture-based controls.
   - **Anomaly Detection**: Implement algorithms to identify unusual or unexpected motions in real time, useful in security systems or fall detection in elderly care.

6. Handle Multi-Person Scenarios
   - Improve the model's ability to handle multi-person scenarios by integrating top-down or bottom-up approaches (e.g., OpenPose or CenterNet) and training on datasets like PoseTrack or CrowdPose.

7. Address Ethical and Privacy Concerns
   - Anonymization: Ensure the model protects user privacy by anonymizing pose data and preventing misuse.
   - Bias Mitigation: Address biases in pose estimation caused by lack of diversity in training datasets (e.g., ethnicity, age, body types).

8. Expand Deployment Scenarios
   - Deploy the model in:
     - Wearables (e.g., smartwatches or fitness trackers).
     - Smart Cameras for surveillance or gesture-based control.
     - Mobile Apps for fitness tracking or gaming.

By implementing these improvements, the human pose estimation model can achieve higher accuracy, robustness, and applicability across a broader range of use cases.

“The project isn’t just about detecting poses, it’s about unleashing potential—turning motion into meaning, data into decisions, and challenges into opportunities across the spectrum of human life and technology.”

REFERENCES

[1]. Ming-Hsuan Yang, David J. Kriegman, Narendra Ahuja, “Detecting Faces in Images: A Survey”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume. 24, No. 1, 2002.

[2]. Patil, Vaibhav, et al. "Gesture Recognition for Media Interaction: A Streamlit Implementation with OpenCV and MediaPipe." *International Journal for Research in Applied Science & Engineering Technology (IJRASET)* (2023).

[3]. Arunnehru, J., Nandhana Davi, A.K., Sharan, R.R. and Nambiar, P.G., 2020. Human pose estimation and activity classification using machine learning approach. In *Soft Computing and Signal Processing: Proceedings of 2nd ICSCSP 2019 2* (pp. 113-123). Springer Singapore.

[4]. Singh, A.K., Kumbhare, V.A. and Arthi, K., 2021, June. Real-time human pose detection and recognition using mediapipe. In *International conference on soft computing and signal processing* (pp. 145-154). Singapore: Springer Nature Singapore.

[5]. Amrutha K, Prabu P, Paulose J. Human Body Pose Estimation and Applications. In2021 Innovations in Power and Advanced Computing Technologies (i-PACT) 2021 Nov 27 (pp. 1-6). IEEE.

[6]. Josyula R, Ostadabbas S. A review on human pose estimation. arXiv preprint arXiv:2110.06877. 2021 Oct 13.