**Project Report**

**ITCS 6166 – Computer Communications and Networks**

**3D Human Modeling**



**Team members**

**Praneeth Reddy Kanchanakuntla**

**Akhil Challa**

**Kousik Varma Dandu**

**Durga Sri Jonnalagadda**

**Contents**

|  |  |
| --- | --- |
| **sno** | **Chapter** |
| **1** | **Introduction** |
| **2** | **Installation Guide** |
| **3** | **Program Details** |
| **4** | **Challenges Faced** |

**code:** [pkanchan15/3D\_Human\_Modeling\_CCN (github.com)](https://github.com/pkanchan15/3D_Human_Modeling_CCN)

**Introduction**

Three-dimensional (3D) human modeling is the process of creating virtual representations of the human body using computer software. This technology has seen widespread use in various industries, including film and video game production, medical research, and fashion design. The use of 3D modeling has revolutionized the way human beings are represented in various media, providing unprecedented levels of realism and detail.

One of the primary uses of 3D human modeling is in the entertainment industry. Video game developers and filmmakers often use 3D models to create realistic and lifelike characters that can interact with the digital world around them. These models can be animated and manipulated to create compelling and engaging content that can transport audiences to new and exciting worlds.

In addition to its use in entertainment, 3D human modeling has also proven to be a valuable tool in medical research. Researchers can use 3D models to study the human body in detail, allowing them to gain a better understanding of how the body functions and how diseases can impact its various systems. This technology has been particularly useful in the development of surgical procedures, as surgeons can use 3D models to plan and practice surgeries before operating on actual patients.

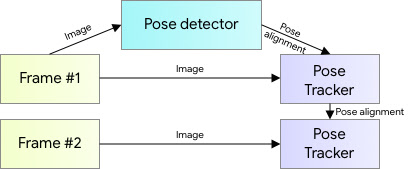
Another area where 3D human modeling is gaining popularity is in fashion design. Designers can use 3D models to create virtual clothing designs, allowing them to see how the clothing will look and fit on a human body before ever creating a physical prototype. This technology has the potential to revolutionize the fashion industry, making it easier and more cost-effective for designers to create new designs and bring them to market.

One of the most popular software tools used in 3D human modeling is Mediapipe. Mediapipe is an open-source framework created by Google that allows developers to build applications that utilize machine learning and computer vision. It provides a robust set of tools for tracking and analyzing human body movements, allowing developers to create highly accurate and detailed 3D models.

Media pipe can be used in a variety of applications, including gesture recognition, augmented reality, and virtual try-on experiences. Its advanced capabilities make it an essential tool for anyone looking to create realistic and lifelike 3D models of the human body.

In conclusion, 3D human modeling has become an essential technology in many industries, providing unprecedented levels of realism and detail in the representation of the human body. It has revolutionized the way we create and interact with digital content, allowing us to explore new worlds and gain a better understanding of the human body. Mediapipe, as an open-source framework, has played a critical role in the development of 3D human modeling, providing a robust set of tools that make it easier than ever to create accurate and detailed models of the human body.

The architecture of media pipe pose detection:

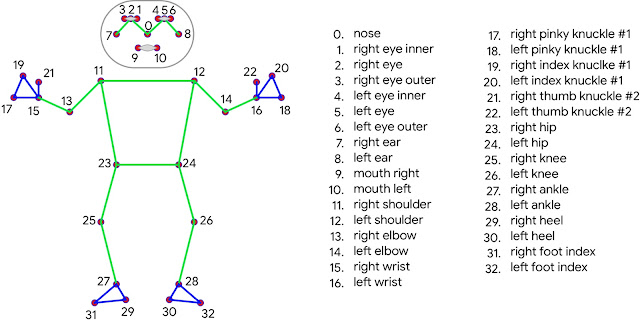


The given video input has taken by frames and the frames are given input to the pose detector of the media pipe and it will return the result that stores the landmarks of the pose from the pose tracker. The landmarks are defined by the Blaze pose which detects around 33 human key points from the body, we have another standard topology called COCO topology which consists of 17 landmarks across the torso, arms, legs, and face. However, the COCO key points only localize to the ankle and wrist points, lacking scale and orientation information for hands and feet, which is vital for practical applications like fitness and dance. The inclusion of more key points is crucial for the subsequent application of domain-specific pose estimation models, like those for hands, faces, or feet.

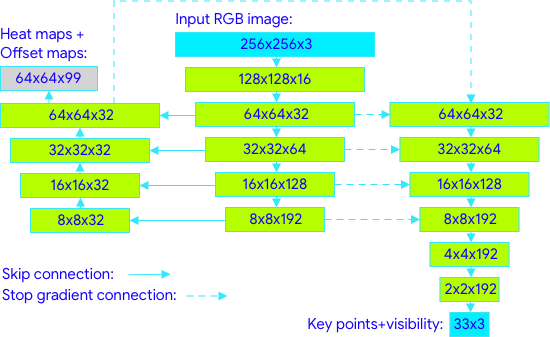
Blaze Pose is a human pose estimation algorithm developed by Google that uses machine learning to accurately detect and track human body movements in real-time. It provides a way to understand human poses and motions in images and video footage, making it an essential tool for various applications, including fitness tracking, sports analysis, and virtual try-on experiences.

Blaze Pose uses a deep neural network to identify key body landmarks, such as joints and extremities and tracks their movements over time. It can detect body movements with high accuracy and speed, making it ideal for real-time applications. Additionally, Blaze Pose can identify multiple people in the same frame and track their movements simultaneously.

One of the significant advantages of Blaze Pose is its ability to work with a wide range of input sources, including video streams, image sequences, and still images. It is also highly efficient, requiring minimal computing resources, which makes it ideal for use in mobile and embedded devices.



The model statistics for the pose estimation, The pose estimation step in the pipeline involves predicting the location of all 33 key points for a person, with each key point having three degrees of freedom, namely its x and y coordinates, as well as its visibility. Additionally, two virtual alignment key points are also predicted. Unlike other methods that use heatmap prediction, which can be computationally intensive, our model uses a regression approach. This regression is supervised by a combination of a heatmap and offset prediction for all key points, as illustrated in the figure below.



The 2d landmarks from the video are obtained from a media pipe and we animate the 2d landmarks of the human body to generate the skeleton of the human body with matching action performed in the uploaded video. The idea is to take the 2d landmarks and convert them into 3d volumetric human figures but it requires external 3d software such as Blender, Maya, or Unity 3d. But all these software are external and computed with huge computational resources and output isn’t deployable in the Web RTC. We used a more advanced algorithm with pre-trained data which draws a perfect outcome.

For the 4th part of the project we used a sophisticated approach called “Video Inference for 3D Human Body Pose and Shape Estimation” Thanks to the authors *Muhammed Kocabas; Nikos Athanasiou; Michael J. Black.* By using their model we can model the 3d Human model. The researchers have created a new type of network structure that incorporates self-attention and have demonstrated that adversarial training at the sequence level can generate motion sequences that are kinematically realistic, even without access to real-world 3D labels for training.

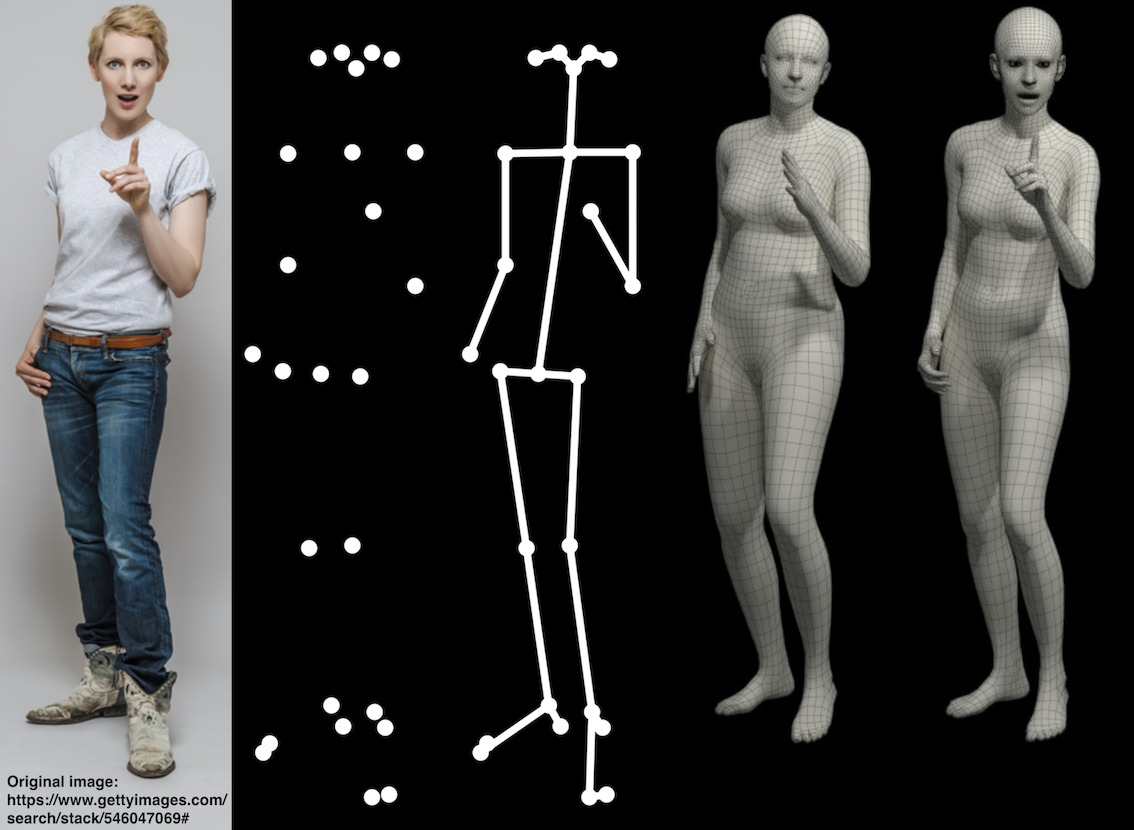
The AMASS (Archive of Motion Capture as Surface Shapes) dataset is a large collection of motion capture data that has been processed and aligned to enable an analysis of the underlying 3D shape of the body. The dataset includes a variety of motions, such as walking, dancing, and sports activities, performed by different subjects of varying body types and ages. In addition to motion capture data, the dataset also includes surface shape information for each frame, allowing researchers to study how the body's shape changes during motion.

The method used in this research is designed to capture the diverse visual appearances of people in images and is validated by using the AMASS dataset to generate realistic human motions. The approach involves using a sequence-based generative adversarial network (GAN) that learns from two sources of unpaired information, allowing it to generate motion sequences that are both visually diverse and kinematically plausible.

In this approach, a temporal model is trained using a video of a person to predict the parameters of the SMPL body model for each frame. The training process also involves a motion discriminator that tries to distinguish between real and predicted motion sequences. By minimizing an adversarial training loss and using the discriminator as weak supervision, the model is encouraged to generate plausible poses that represent realistic motion sequences.

The VIBE model is trained using images captured in natural settings and relies on a combination of pre-trained convolutional neural networks (CNNs), a temporal encoder, and a body parameter regressor to predict SMPL body model parameters. The predicted poses are then evaluated by a motion discriminator, which distinguishes between real and fake motion sequences by comparing them to poses sampled from the AMASS dataset. To capture the sequential nature of human motion, both the temporal encoder and motion discriminator are implemented using Gated Recurrent Units (GRUs).

To summarize, this paper makes several contributions to the field of 3D human motion estimation. Firstly, it uses the AMASS dataset for adversarial training of the VIBE model, which helps to generate more realistic and accurate motion sequences. Secondly, an attention mechanism is employed in the motion discriminator to better weigh the contribution of individual frames, resulting in improved performance compared to baseline approaches. Thirdly, the paper provides a quantitative comparison of various temporal architectures for 3D human motion estimation. Finally, the proposed approach achieves state-of-the-art results on major 3D pose estimation benchmarks.



The project is focused on the above image, initially, the video is uploaded in the Streamlit web rtc, and the input is taken by using the 2d landmarks from the frames obtained from the video we get the second phase of the image, and by using the nb\_helpers class to connect the landmarks and show the output in a video. The Vibe program allows us to model the video to show the output as shown in Figures 4 and 5.

**Installation Guide:**

Software requirements for the project are:

1. Mini Conda

2. Nivida Graphics

3. Dependencies (from requirements.txt)

The running of the project:

The project contained two files to run

1. app.py

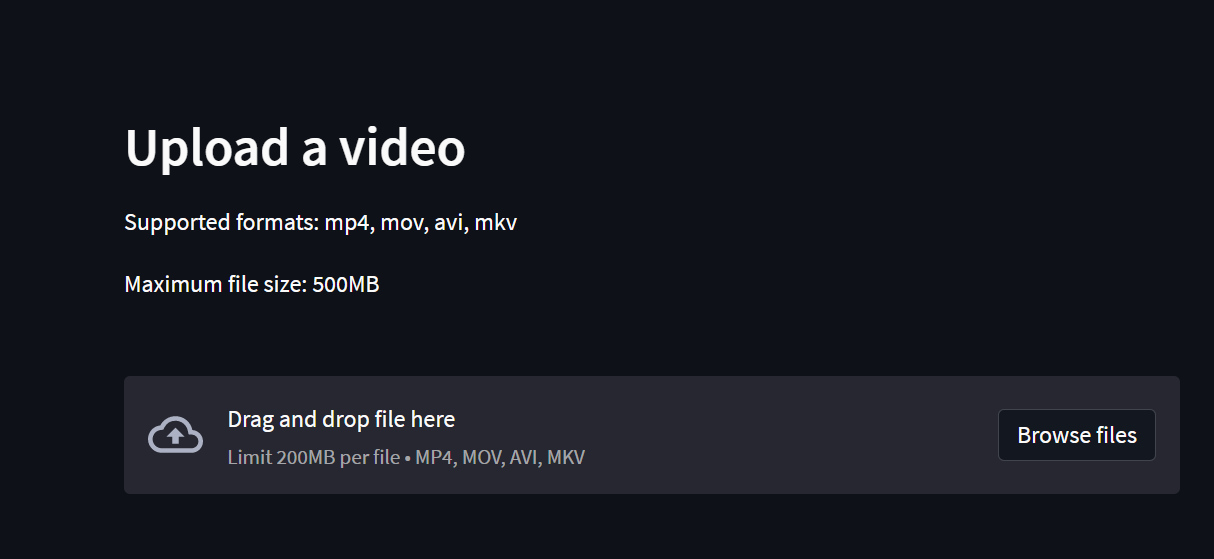
2. demo.py

The program “app.py” display a streamlit web app that asks the user to upload the video to convert the video to display the video with extracted landmarks. The program displays the skeleton of the human.

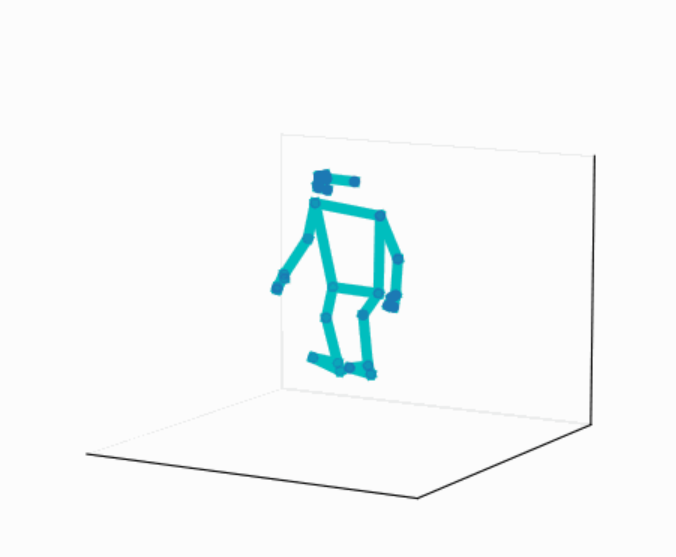
The program “demo.py” is a vibe (Video Inference for 3D Human Body Pose and Shape Estimation) module that converts the video, to a 3d human model by using the dataset and pytorch models.

VIBE uses CNNs to extract image features. The output from the CNN is fed as input to the recurrent neural network, which processes the sequential nature of human motion. Then a temporal encoder and regressor are used to predict the body parameters for the whole input sequence.

The picture below describe the function which accepts the video as input and take input as frames and give that frames to mediapipe pose detection to obtain the landmarks of the actor and it passes these landmarks to time\_animate function of nb\_helpers class, this helps to animate the landamarks to video.

To run the file run the following command **streamlit run app.py** and it will show the following run at the local host: <http://localhost:8501/>, the following the output at localhost. 

The uploaded file is saved in the upload folder and the output is shown in the web page



That is the output of the program it is the same skeleton image shown in the above image.

The demo.py shows the output,

We have run the following code:

python demo.py --vid\_file third.mp4 --output\_folder output/ --sideview

we would like to acknowledge the authors of the paper and also Muhammed Kocabas for the pre-existing code for easy access.

The output of the following program is a 3d animated figure which overlays the real video.



Installation for project 🡪 vibe

!git clone <https://github.com/mkocabas/VIBE.git> (vibe project)

%cd VIBE/

# Install the other requirements

!pip install torch numpy==1.17.5

!pip install git+https://github.com/giacaglia/pytube.git --upgrade

!pip install -r requirements.txt

!source scripts/prepare\_data.sh

!python demo.py --vid\_file third.mp4 --output\_folder output/ --sideview

# Play the generated video

from IPython.display import HTML

from base64 import b64encode

def video(path):

  mp4 = open(path,'rb').read()

  data\_url = "data:video/mp4;base64," + b64encode(mp4).decode()

  return HTML('<video width=500 controls loop> <source src="%s" type="video/mp4"></video>' % data\_url)

video('output/third/third\_vibe\_result.mp4')

to display the video file with 3d human model animation which overlays the original video which shows the output as shown in the given figure.

**Challenges Faced**

The Challenges faced while running the project are,

1. We faced a problem while handling the 2d landmarks obtained from the media pipe pose detection library and converting them into 3d human models.

2. To facilitate the 3d model, we used the VIBE project to display the output in a 3d human model.

3. we tried different 3d modeler software such as Blender, Maya, or Unity, but we are unable to incorporate the output in web rtc.

4. Many different approaches are there to model the video to a 3d human model, but many programs require installation errors.

5. The vibe model requires GPUs, rather I we used google collab to run the program.

­­­­­­