Computer Communication & Networks

Project Report

HumanNeRF: Free-viewpoint Rendering of Moving People from Monocular Video (CVPR 2022)

Team 12:

Rithvik Reddy Patel
Neeraj Chandel
Rahul Yadav
Hinduja Bikki

Introduction

In tackling the formidable challenge of free-viewpoint rendering from monocular videos capturing dynamic human activities, this project introduces HumanNeRF, a novel method designed to revolutionize the rendering landscape. The conventional hurdles of single-camera "in-the-wild" videos, with their complex body poses and non-rigid motions, have posed persistent obstacles for existing neural rendering techniques. HumanNeRF offers a groundbreaking solution by utilizing a single video input, employing per-frame segmentation and automated 3D pose estimation, and optimizing for a canonical volumetric T-pose with a motion field that encompasses both skeletal rigid and non-rigid volumetric representations. This data-driven approach, devoid of template models, enables unparalleled flexibility in rendering. Whether applied to lab datasets, real-world videos, or YouTube content, HumanNeRF excels numerically and visually, allowing for on-the-fly pausing at any frame and rendering from diverse viewpoints, including a full 360-degree camera path. The project showcases significant advancements in free-viewpoint rendering, producing superior results in capturing the intricate details of human motion and appearance.

Free-viewpoint rendering, a complex task involving geometry and surface property modeling for rendering from new camera perspectives, faces challenges in recreating intricate geometry and subtle lighting effects. Existing methods, such as image-based rendering, have been explored extensively over the years. Human-specific rendering, historically investigated by pioneers like Kanade et al., utilized domes and silhouettes to recover depth maps, while subsequent works integrated parameterized body models and marker-less motion capture. Traditional methods

rely on multi-view setups, often in studios, whereas this project focuses on a simpler monocular camera configuration. Comparisons are drawn with dynamic and deformable Neural Radiance Fields (NeRF), while human-specific neural rendering approaches, including Liu et al. and Martin-Brualla et al., leverage pre-captured body models and UNet training. The proposed method distinguishes itself by accepting monocular videos with complex human motions, achieving high-fidelity 3D rendering, and surpassing similar approaches in capturing pose-dependent, non-rigid deformation for free-viewpoint applications. The formulation draws inspiration from Vid2Actor but excels in free-viewpoint scenarios, outperforming concurrent works such as Xu et al., Su et al., and Noguchi et al.

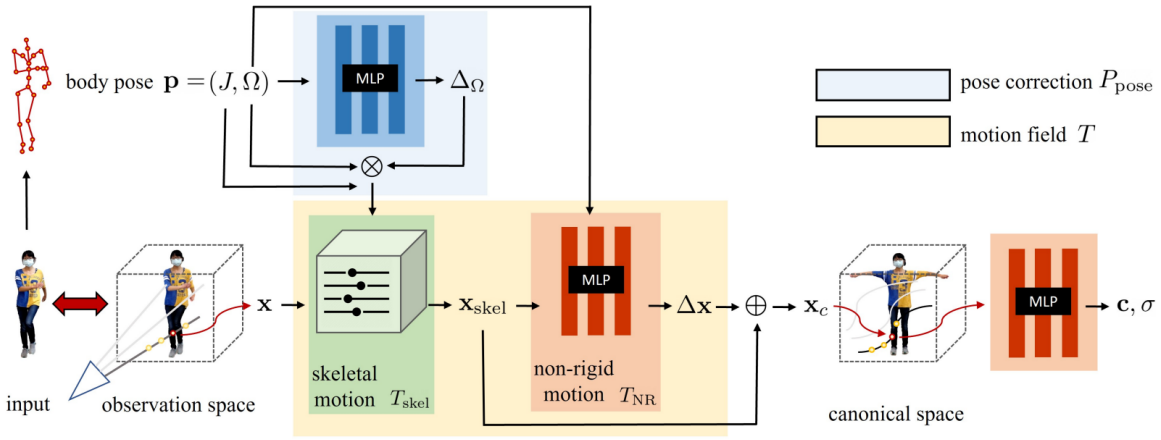
Working

This report explores three powerful approaches, each with unique strengths:

1. Posing the Canonical: This approach envisions a single, "canonical" representation of the human body in a standard pose. A neural network then manipulates this representation to create diverse poses. Its strength lies in its simplicity and ability to handle a wide range of poses. However, its expressiveness is limited, struggling with complex deformations like clothing folds and potentially losing fine details during pose changes.

2. Skeleton and Beyond: This approach deconstructs the motion field into two components: a skeleton-driven component capturing coarse deformations through standard skinning techniques, and a non-rigid component modeled by a separate neural network that captures finer details and clothing effects. This enables high-fidelity results with accurate representation of intricate movements and clothing interactions. However, it necessitates a more complex system with additional computational requirements and relies heavily on sufficient training data for the non-rigid network.

3. Efficient Primitives: This approach represents the human body as a collection of individual volumetric primitives, each equipped with its own neural network for color, density, and pose deformation. This allows for efficient handling of large and complex models, enabling parallel processing of individual primitives. However, it requires careful design and training of each primitive to avoid potential artifacts at the boundaries between them.



The image shows a neural field architecture for human representation that is both efficient and accurate. It first divides the human body into a set of volumetric primitives. Each primitive is then represented by a neural network that maps from position to color and density. The neural networks for each primitive are trained separately, and then they are combined to form a single neural network that represents the entire human body. To render the human body, the system first predicts the pose of each primitive. This is done using a standard skeleton-driven algorithm. The predicted pose is then used to deform the neural network for each primitive. The deformed neural networks are then combined to produce a final neural network that represents the entire human body in the desired pose.

The neural network for the entire human body can then be used to render the human body at any desired resolution. To do this, the system simply samples the neural network at the desired resolution and outputs the color and density at each sample point. This neural field architecture has several advantages over other approaches for human representation. First, it is very efficient. This is because the neural networks for each primitive can be trained separately and then combined to form a single neural network. This makes it possible to train the system on vast datasets of human bodies.

The optimization process is based on minimizing a loss function that measures the difference between the rendered image and the ground truth image. The optimization process is iterative. At each step, the neural network weights are updated to minimize the loss function. The optimization process is stopped when the loss function reaches a minimum. The Optimizing HumanNeRF part of the image also shows how the system deals with non-rigid deformations. Non-rigid deformations are deformations that cannot be modeled by a linear transformation. Clothing folds and facial expressions are examples of non-rigid deformations. The system models non-rigid deformations using a neural network. The neural network is trained to predict the non-rigid deformation of each primitive based on the estimated pose of the primitive. The predicted non-rigid deformations are then used to deform the neural networks for each primitive. The deformed neural networks are then combined to produce a final neural network that represents the entire human body with non-rigid deformations. The optimization process then

optimizes the weights of the neural network to minimize the loss function, taking into account the non-rigid deformations.

Implementation

Project Setup:

- Uploaded the Jupyter Notebook ("HumanNeRF.ipynb") to Google Colab.
- Configured runtime settings for optimal performance, including resource allocation (GPU/TPU) and Python version (if needed).
- Executed the code cells, either individually or all at once.
- The code automatically installed necessary libraries.

Results Generation:

Generated three video outputs showcasing different capabilities:

- *Freeview.mp4:* Demonstrates free-viewpoint rendering of a moving person.
- *Movement.mp4*: Focuses on the natural motion capture and rendering of the subject.
- *T-pose.mp4*: Displays the ability to render the subject in a standard T-pose.

Webpage Development:

- Built a web application leveraging Streamlit and Flask for efficient and interactive video streaming.
- *Streamlit server:*
  - Responsible for displaying the video using the Streamlit library.
  - Functions as the backend for the application.
- *Flask server:*
  - Serves the video content as a stream for seamless playback.
- *HTML client*:
  - Embeds the video player using the HTML video element.
  - Sources the video from the Flask server.
  - Offers user controls for playback and interaction.

Webpage Deployment:

- Installation of Streamlit prerequisite.
- Navigation to the webpage directory in Terminal.
- Execution of the "streamlit run server.py" command to launch the application.
- Accessing the webpage ("client.html") to view the video demonstration with controls.

Outcomes

- Successful replication of the original HumanNeRF research, allowing for free-viewpoint rendering of moving people from a single video source.
- Development of a user-friendly webpage showcasing the project's capabilities through three distinct video demonstrations.
- Implementation of a robust and efficient streaming solution using Streamlit and Flask, ensuring smooth playback and user interaction.
- Creation of an accessible interface through the HTML client, allowing users to control the video playback and gain a deeper understanding of the project's functionalities.

Challenges

Environmental Challenges: The project repository is tailored for Linux environments and assumes the use of four GPUs by default. This poses difficulties when deploying the code on older local machines with limited resources.

Compatibility Hurdles: The project relies on state-of-the-art libraries and frameworks, leading to potential compatibility issues on older systems or with different versions of required dependencies.

Tedious Rendering Process: The neural field rendering process can be time-consuming, particularly on machines with restricted resources.

Google Colab Runtime Limitation: Google Colab imposes a 12-hour runtime limit for notebooks, which can be problematic when dealing with intricate neural field models.

Size Constraints of the Repository: The project repository is substantial in size, making it challenging to deploy on cloud platforms with storage limitations.

Cloud Platform Deployment Challenges: Deploying neural field models on cloud platforms can be intricate, especially for complex models with multiple dependencies.

Infeasibility for Streaming Platforms: This project is unsuitable for streaming platforms due to its prolonged processing time and uncertainties in input formats.