Replication of Postures onto 3D Models Utilizing Machine Learning

Christian Narcia

Aram Flores

Moises Cervantes

christian.narcia01@utrgv.edu, aram.flores01@utrgv.edu, and moises.cervantes01@utrgv.edu

10/8/2020

# Summary of the Proposal

Manually posing a 3D model is, at best, mildly time-consuming, but becomes exponentially tedious depending on the model itself and its overall complexity. However, if a program was taught to recognize the various macro appendages of the human body based on an image, then identify the pose, it could then be trained to replicate the analyzed pose on a 3D model. This will, firstly, require the program to be fed macro-appendage data (via Supervised Learning) and properly trained on its identification accuracy given a human body. With the knowledge gained previously being utilized as a basis, it must construct a Human Pose Skeleton which will then be utilized in 3D software to correctly pose the 3D model. This could be implemented as an add on tool for Blender for inspiring creators to use.

# Background

In any career or field of study that requires one to work with 3D software (be it for modeling, animation, video game development, etc.) the act of constantly rigging and posing models into a desirable position is common and a crucial process. As such, countless hours are reserved for such a meticulous task, and, as such, the reality of it being prone to error is evermore present (via work fatigue). Motion and Video Tracking is the closest references to our subject despite not really seeing an implemented form where the tracking is moving the 3D character itself. Animoji is another reference, but it is only limited to face features. There is also the Xbox Kinect, that already has implemented motion tracking and requires an actor in front of the Xbox Kinect Camera. However, the difference is in the dataset: the Kinect uses a constant video feed which, in turn, provides a constant reference feed, and must dynamically make the pose. We know that the program needs to recognize and label the various limbs of the human body to create the mapping of the body.

We decided that the implementation of Supervised Learning was the correct approach to this step, as the dataset target to be taught was not unfathomably large and the allocation of additional time, where-ever needed, to syphon the correct data into the program was deemed tolerable and necessary nevertheless. It must also be able to properly implement a mapping of the given pose into a Human Pose Skeleton. Initially, we thought of giving our program the capabilities of being able to analyze and construct multiple Human Pose Skeletons from a single image, but we soon realized that such concept would require much more intricate implementations that far exceed our current goals of achieving the same effect on a single human pose, so we decided to focus on one human per image. In terms of the 3D software, we choose Blender because a few team members are familiar with this program and can create a simple rigged 3D model. When it comes to merging Python and Blender, Blender has integrated python within itself, so even if any issues arise with data not being compatible with Blender, an easy fix is to import it from our Integrated Development Environment (Pycharm) via text files. As for the images’ dimensions, we will utilize a program that standardizes them to a manageable size.

This is our expectations on how our ML code should correctly Identify limbs.

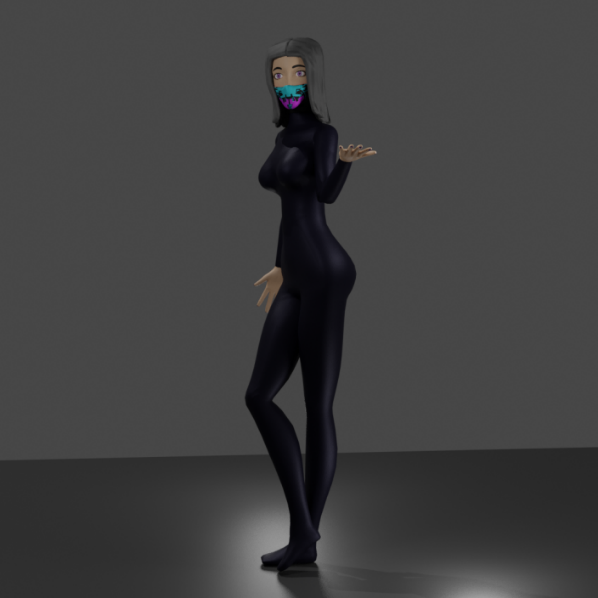


Image on the right is an example of labels, and below is the expected output

* Our challenge is making a Machine Learning program that can get any pose, attach it to the 3D model, then another completely different pose/image, and apply that pose to the same 3D model.

# Goal and Objectives

The main goal of our research is to utilize Machine Learning to process the pose of a human (given an image) and to then replicate it into a 3D model utilizing Blender and Python. To achieve this, our first objective is to devise a way for the program to read the data (images) fed to it and identify the anatomy of the model. This will be implemented via Supervised Learning in which the labels of each limb will be provided. The second objective is to then scale the acquired data into a Human Pose Skeleton for pose mapping, which will be implemented by converting the data analyzed from the image into a dataset. Of course, to use a 3D model, one would need to create a 3D model, so our third objective is to make and rig a 3D model. The fourth objective is to incorporate the data with Blender so that a 3D model can be replicated. This will be achieved by providing the necessary parameters for each limb (coordinates, rotation, position, etc.).

# Data and Methods

The data we will use (for both the training of appendages and the testing of human images) will be extracted from various online sources. For the testing data, it will consist of full body images of people. For the training data of the appendages, the program will utilize images that will emphasize of the said appendage. The data size can be as many images as needed, possibly reaching 1000 .png files. These images can be standardized to be 512 pixels \* 512 pixels. We can implement different classes/features, to help with the visualization and categorization of the macro-appendages. We will use a program to fetch and gather images from Shutterstock images for our data set. We will use dataset files to store our values required for Blender to pose the 3D model. We could run a filter that helps emphasize human features and filter out unnecessary background. This could be helping our data, making clearer and more concise datasets. We could use a face recognition program to aid the process in helping identify which direction the model will be facing and where to begin the Human Pose Skeleton. Then the program will have a guarantee reference point, where the limbs can start being identified on the body.

**REFERENCES**

<https://www.awn.com/news/optitrack-releases-motive-20-motion-tracking-software>

<https://blender.stackexchange.com/questions/140795/does-blender-have-full-body-motion-tracking?noredirect=1&lq=1>

[https://www.kdnugget• Kinect SDK: https://www.microsoft.com/en-us/downl...](https://www.kdnugget•  Kinect SDK: https://www.microsoft.com/en-us/downl...Delicode's NI Mate & Add-on: https://ni-mate.com/Remington's NI Mate Rig: https://remington.pro/resources/asset...s.com/2019/06/human-pose-estimation-deep-learning.html)

[Delicode's NI Mate & Add-on: https://ni-mate.com/](https://www.kdnugget•  Kinect SDK: https://www.microsoft.com/en-us/downl...Delicode's NI Mate & Add-on: https://ni-mate.com/Remington's NI Mate Rig: https://remington.pro/resources/asset...s.com/2019/06/human-pose-estimation-deep-learning.html)

[Remington's NI Mate Rig: https://remington.pro/resources/asset...s.com/2019/06/human-pose-estimation-deep-learning.html](https://www.kdnugget•  Kinect SDK: https://www.microsoft.com/en-us/downl...Delicode's NI Mate & Add-on: https://ni-mate.com/Remington's NI Mate Rig: https://remington.pro/resources/asset...s.com/2019/06/human-pose-estimation-deep-learning.html)

<https://realpython.com/face-recognition-with-python/>