CV Project: Human-object interaction with Optitrack

Giovanni Lorenzini
223715
giovanni.lorenzini@studenti.unitn.it

Davide Dalla Stella
223727
davide.dallastella@studenti.unitn.it

Abstract

In this work a human-object MoCap was implemented. In order to do that a 3D model of an object was obtained by using a Neural Radiance Field, especially 3 different networks were tested and compared. Once the 3D model was obtained it was imported into the graphics engine Unreal Engine and made it to interact with a meta-human through the use of OptiTrack and Motion Builder softwares.

The source code can be found here: https://github.com/lorenzinigiovanni/cv-project.

1. Introduction

The goal of this project is to asses the interaction between a human and an object using OptiTrack and Unreal Engine. The project is divided in two parts:

1. extract a 3D model of the object through its photos;
2. make the model interacts with a human model.

The chosen object to extract it's 3D model is a chair. To be able to generate it, it was necessary to acquire photos of it and use a neural network to process them.
In total, 3 different neural networks were tested, all based on the NeRF method

2. Object model generation

2.1. NeRF

Neural Radiance Field [3] is a method that optimize a fully-connected neural network without any convolutional layer. This network is capable of generating 3D views starting from a set of 2D images of a scene. The input is a 5D vector coordinate containing the information about the spatial location (x, y, z) and the view orientation (θ, ϕ) while the output represents the volume density and the view-dependent emitted radiance at that spatial location.

$$(x, y, z, \theta, \phi) \longrightarrow F_\Theta \longrightarrow (RGB\sigma)$$

Since this process is naturally differentiable it uses the gradient descent to optimize the model by minimizing the error between the observed images and the corresponding views rendered. So this method can represent complex real-world geometry and appearance and are well suited for gradient-based optimization using projected images like volumetric representation and overcomes the prohibitive storage costs of discretized voxel grids when modeling complex scenes at high-resolutions.

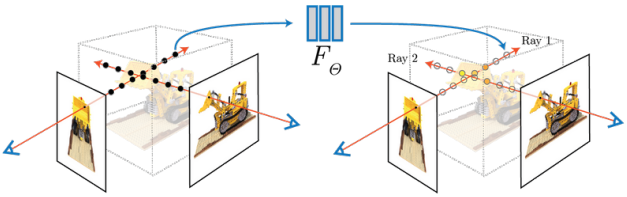

Figure 1. NeRF's procedure.

2.2. pixelNeRF

The initial idea was to use 8 input photos acquired by the OptiTrack cameras. As the number of images is low, the natural candidate technique to use is pixelNeRF [14], that should works well with few images, even a single one.

We used the code from [10] with some slightly changes to adapt it to our data. In particular this is how the process works:

1. load the images;
2. use detectron2 [13] to segment the chair and eliminate the background;
3. train the network;
4. obtain novel views of the object.

A drawback of this method is that the pose of each input image is necessary but it can be quite challenging to obtain good estimates of them. A solution is to use Colmap but it cannot extract the poses given our images, so we keep the

same poses found by the authors of [10] since the cameras we used are the same of the same laboratory.

The results we obtained are quite poor, due to the fact that the ROI in the images was quite small as was the resolution and probably also due to the low light.

In Fig. 2 a qualitative result of what we obtained using pixelNeRF can be seen.

Figure 2. Chair model obtained using pixelNeRF.

2.3. BARF

Since a limitation of pixelNeRF is having accurate camera poses, we looked at BARF (Bundle-Adjusting Neural Radiance Fields) [1] which is a technique that works with unknown camera poses.

Even with this technique the obtained results are poor, probably due to the fact that the number of input images is very small.

In Fig. 3 a qualitative result of what we obtained using BARF can be seen.

2.4. NeRF-PL

This unofficial implementation of NeRF aims to provide a simpler and faster training procedure. To train this network it's necessary to use between 40 and 60 photos for a 360 degree model and use Colmap [11, 12] with a sparse reconstruction in order to obtain the views of the cameras of the object. LLFF [2] was used to construct the dataset. For the purpose of this project 50 images of the chair was acquired using an iPhone with the rear high resolution camera.

Figure 3. Chair model obtained using BARF.

When we tried this network for the first time we did not have enough images available to try to extract the model of the same chair used for the other networks, for this reason we initially photographed one of our personal chairs (Fig. 4) and only after observing the results we went to the lab to acquire more data.

The network was trained by using the code in [8] using a NVIDIA RTX 3080 for about 6 hours.

At the end we extracted the mesh of the object using a script from [8]. The results are very good as it can be seen in Fig. 5.

2.5. Instant Neural Graphics Primitives

To speedup the training fase and obtain better results we choose to try Instant-NGP [4]. The code that we used is from [9] that uses PyTorch [7].

The drawback of this technique is that a GPU with a lot of VRAM and a PC with a large amount of RAM are needed. We were not able to run it on our PCs. As a future work, it can be interesting to use it to improve the generated mesh.

Figure 4. Personal chair model obtained using NeRF-PL.

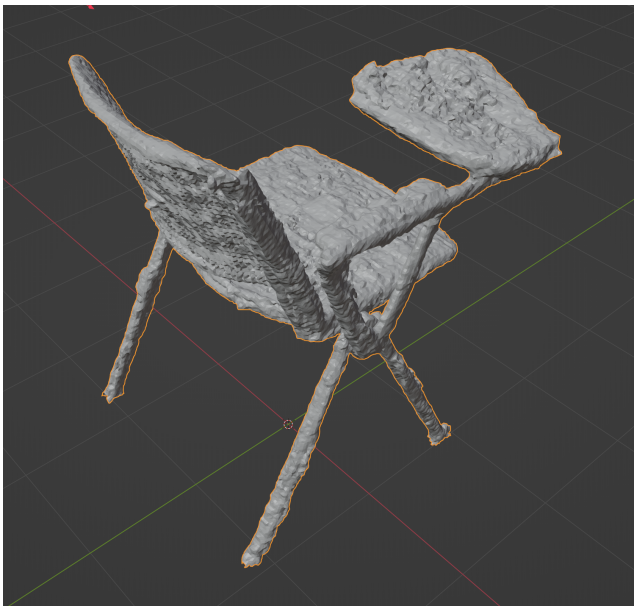

Figure 5. Extraced 3D model using NeRF.

3. Human-object interaction

3.1. OptiTrack

The OptiTrack motion capture system we used is optical. The acquisition system is composed of 8 OptiTrack Prime^x 22 cameras that provides a tracking accuracy of $\pm 0.15mm$ [6]. What the cameras "see" are the retroreflective markers that needs to be attached to the object and to the human body to track. On the chair 15 markers has been attached taking care to not create symmetrical patterns. The suite worn by the human has 36 marker attached.

The OptiTrack Prime^x 22 cameras have on board an image processing device [6] to recognize the reflective markers. Next the Motive software elaborate the information from all the cameras to compute a 3D position for each marker. It's then possible to create rigid bodies (like a char) or skeleton (like a human) that can be transmitted to Unreal Engine.

3.2. Chair only movement

As a first test only the movement of the chair was tested. The procedure that we followed is the following:

1. The model generated using NeRF-PL has been imported to blender were some noise and other meshes has been removed.
2. A blank project in Unreal Engine has been created and then the model has been imported.
3. The model has been scaled in realistics dimensions as NeRF does not create it in a 1:1 scale.
4. We chose to use the OptiTrack Streaming Client Plugin [5] that needs to be installed inside Unreal Engine.
5. In Motive a rigid body with the markers of the chair has been created and its streaming enabled.
6. In Unreal Engine the chair has been linked with the data stream from Motive.

As it's possible to see in Fig. 6 the chair inside Unreal Engine is moved by using the real life position acquired by the OptiTrack cameras.

Figure 6. Chair in Unreal Engine moved by OptiTrack data.

3.3. Human chair interaction

At the end, the interaction between a human and the chair in Unreal Engine was tested using the real data provided by OptiTrack.

By using the OptiTrack Streaming Client Plugin in Unreal Engine we were not able to animate the skeleton (due to a plugin bug) and so it was necessary to pass first the data from Motive to MotionBuilder and then to OptiTrack.

The procedure that we followed is the following:

1. In Motive a skeleton with the markers of the human has been created and its streaming enabled.

2. In MotionBuilder the plugin to connect to Motive has been activated and configured to receive the data.

3. Then the parts of the body has been associated with the skeleton.

4. Using the plugin to connect to Unreal Engine the skeleton and chair position has been exported.

5. In Unreal Engine a meta human has been downloaded and a animation blueprint created and configured to use the streaming data from MotionBuilder.

As it's possible to see in Fig. 7 the human interacts with the chair by sitting on it and writing on its table.

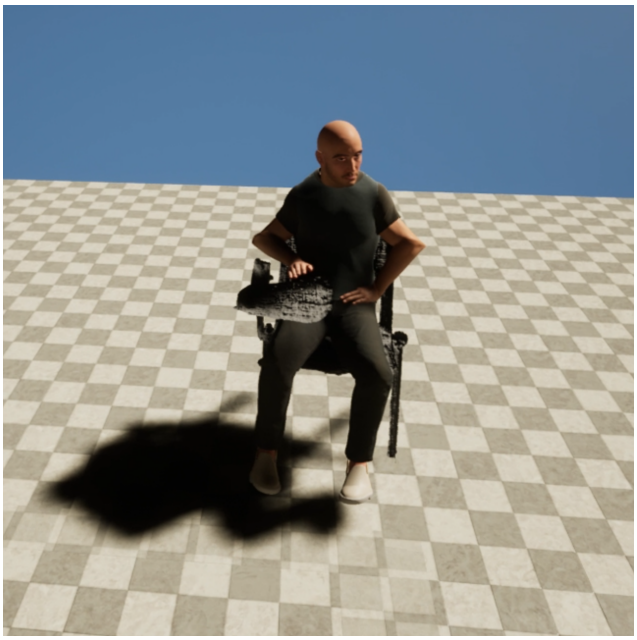

Figure 7. Human and chair interaction in Unreal Engine moved by OptiTrack data.

4. Conclusions

The results obtained, although satisfactory, are not the best. Among the various neural networks tested, the best result obtained is certainly that of the NeRF-PL, although it requires a larger number of images than the other networks. The aim of this project was to be able to extract a 3D model and then use it in a MoCap process. This can be considered achieved but for MoCap applications it is clear that there are more efficient methods to obtain a model that can be used. Using NeRF is generally very expensive in terms of resources and computation time, moreover it is necessary to pay close attention that the photos of the model to be given as input reproduce all the angles correctly and that they are also of good quality.

5. Future works

We thinks that to improve this project the followings can be done:

- Use Instant Neural Graphics Primitives to generate the mesh. A sensible inferior execution time and a slightly improvement of the results has to be expected.

- For the chair model use as the center of the mesh the same center that is found in Motive. This will improve the tracking.

- Scale the skeletons to match the real size of the skeleton acquired in Motive.

References

[1] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 2

[2] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019. 2

[3] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1

[4] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. 2

[5] OptiTrack. Optitrack streaming client plugin. https://docs.optitrack.com/plugins/optitrack-unreal-engine-plugin/unreal-engine-optitrack-streaming-client-plugin. 3

[6] OptiTrack. Optitrack website. https://optitrack.com/. 3

[7] PyTorch. Pytorch source code. https://github.com/pytorch/pytorch. 2

[8] Chen Quei-An. Nerf_pl: a pytorch-lightning implementation of nerf, 2020. 2

[9] Chen Quei-An. Ngp_pl: a pytorch-lightning implementation of instant-ngp, 2022. 2

[10] Thomas Reolon and Moreno D'Incà. P-nerf source code. https://github.com/thomasreolon/P-NeRF. 1, 2

[11] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[12] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[13] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 1

[14] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 1