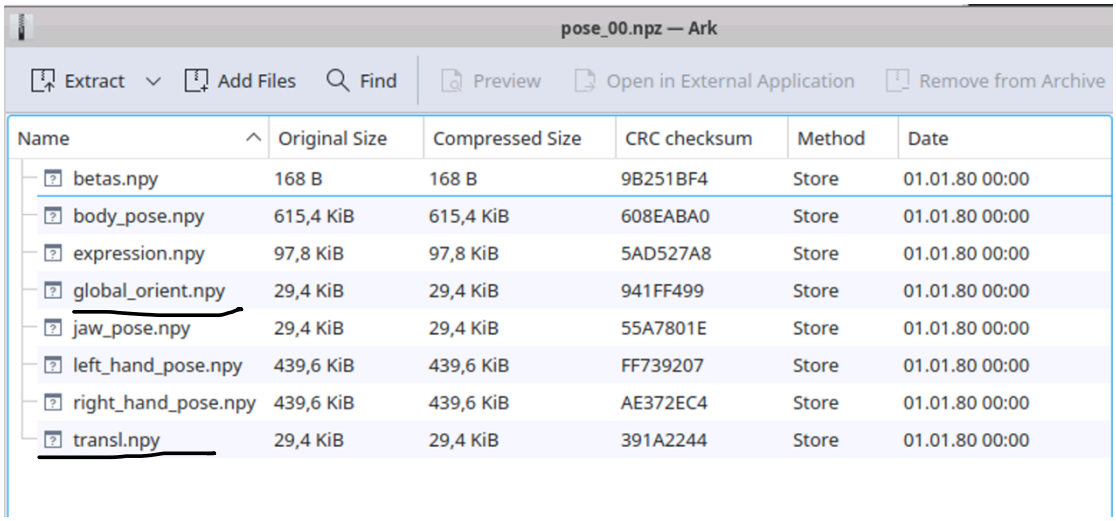
有可能会遇到的问题

情况一：替换不同的动作序列导致的已经匹配好的人物位置不同

Solution：

First step：将需要匹配的动作序列（npz 文件）中的 transl 和 global_orient 替换成 /home/zhiyw/Desktop/AnimatableGaussians/configs/thuman4/pose_00.npz 中的 transl 和 global_orient



Second step：使用新生成的 npz 文件训练人物高斯

Run（python main_avatar.py -c configs/人物文件夹/avatar.yaml --mode=test）

```
test:
  dataset: MvRgbDatasetAvatarReX
  data:
    data_dir: /home/zhiyw/Desktop/AnimatableGaussians/configs/zzr
    frame_range: [0, 1]
    subject_name: avatarrex_zzr
  pose_data:
    data_path: /home/zhiyw/Desktop/AnimatableGaussians/configs/thuman4/01_modified.npz
    frame_range: [0, 1]
#    data_path: Z:/Data/Pose/AMASS/CMU/06/06_13_poses.npz
#    data_path: Z:/Data/Pose/AMASS/CMU/10/10_05_poses.npz
#    frame_interval: 4
```

Figure1：使用新 npz 文件（01_modified.npz）训练，avatar.yaml 的配置

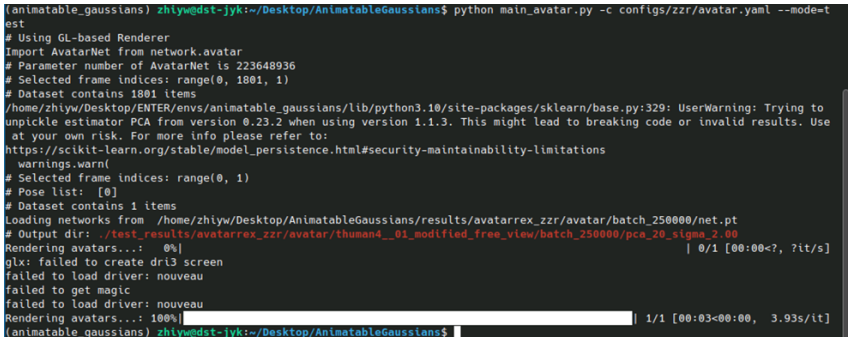


Figure2：训练动作序列

Step 3：将训练好的 ply 文件进行位置转换（应用 trans matrix）

Trans matrix 的文件在/home/zhiyw/Desktop/animatable_dataset/human_trans/trans.md

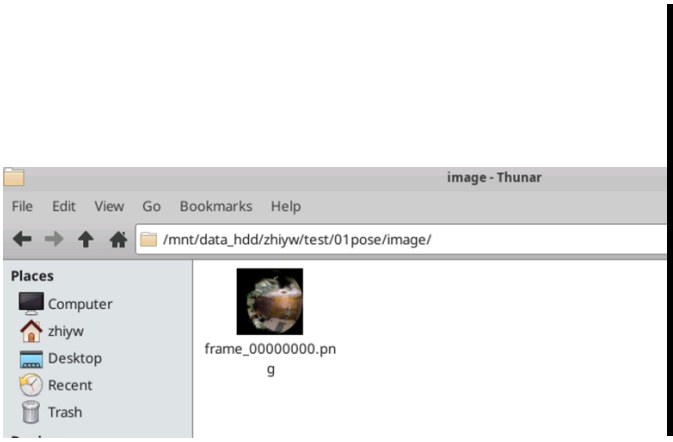
```
(hugs) zhiyw@dst-jyk:~/Desktop/ml-hugs$ python scripts/transform_human_sequence.py  --input_dir /mnt/data_hdd/zhiyw/test/01pose/original/ply   --output_dir /mnt/data_hdd/zhiyw/test
/01pose/djr   --output_format pt --start_frame 0  --end_frame 0       2026-01-09 16:55:04.822 | INFO     | __main__:main:292 - Processing frames 0 to 0
2026-01-09 16:55:04.822 | INFO     | __main__:main:293 - Input directory: /mnt/data_hdd/zhiyw/test/01pose/original/ply
2026-01-09 16:55:04.823 | INFO     | __main__:main:294 - Output directory: /mnt/data_hdd/zhiyw/test/01pose/djr
2026-01-09 16:55:04.823 | INFO     | __main__:main:295 - Output format: pt
Processing frames:   0%|                                                                                                | 0/1 [00:00<?, ?it/s]2026-01-09 16:55:04.824 | INFO     | __main__:load_ply_
gaussians:24 - Loading PLY file: /mnt/data_hdd/zhiyw/test/01pose/original/ply/00000000.ply
2026-01-09 16:55:10.770 | INFO     | __main__:save_gaussians_to_pt:254 - Saved transformed PT to: /mnt/data_hdd/zhiyw/test/01pose/djr/pt/00000000.pt
2026-01-09 16:55:10.774 | INFO     | __main__:main:322 -  Processed frame 00000000
Processing frames: 100%|████████████████████████████████████████████████████████████████████████████████| 1/1 [00:05<00:00,  5.95s/it]
2026-01-09 16:55:10.774 | INFO     | __main__:main:328 -  All frames processed successfully!

============================================================
TRANSFORMATION SUMMARY
============================================================
Input directory: /mnt/data_hdd/zhiyw/test/01pose/original/ply
Output directory: /mnt/data_hdd/zhiyw/test/01pose/djr
Frames processed: 0 to 0
Output format: pt
Transformation matrix applied:
[[ 0.00450604 -0.12459285  0.00340451 -3.70095587]
 [ 0.14971124  0.00826904  0.00426282 -2.73571181]
 [-0.00813861  0.00311586  0.12460145 -4.24491024]
 [ 0.          0.          0.          1.        ]]
============================================================
```

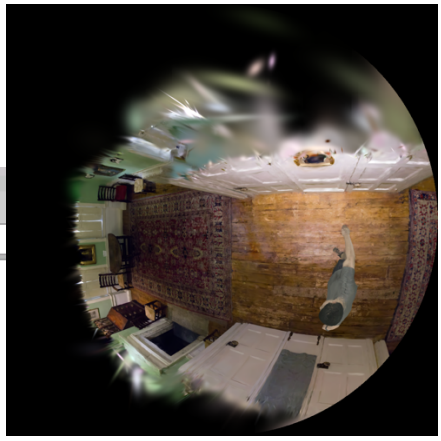
```
(hugs) zhiyw@dst-jyk:~/Desktop/ml-hugs$ python scripts/transform_human_sequence.py  --input_dir /home/zhiyw/Desktop/AnimatableGaussians/test_results/avatarrex_zzr/avatar/thuman4__0
1_modified_free_view/batch_250000/pca_20_sigma_2.00/posed_gaussians   --output_dir /mnt/data_hdd/zhiyw/test/01pose/original   --output_format ply --start_frame 0  --end_frame 0
2026-01-09 16:54:23.576 | INFO     | __main__:main:292 - Processing frames 0 to 0
2026-01-09 16:54:23.576 | INFO     | __main__:main:293 - Input directory: /home/zhiyw/Desktop/AnimatableGaussians/test_results/avatarrex_zzr/avatar/thuman4__01_modified_free_view/b
atch_250000/pca_20_sigma_2.00/posed_gaussians
2026-01-09 16:54:23.577 | INFO     | __main__:main:294 - Output directory: /mnt/data_hdd/zhiyw/test/01pose/original
2026-01-09 16:54:23.577 | INFO     | __main__:main:295 - Output format: ply
Processing frames:   0%|                                                                                                | 0/1 [00:00<?, ?it/s]2026-01-09 16:54:23.578 | INFO     | __main__:load_ply_
gaussians:24 - Loading PLY file: /home/zhiyw/Desktop/AnimatableGaussians/test_results/avatarrex_zzr/avatar/thuman4__01_modified_free_view/batch_250000/pca_20_sigma_2.00/posed_gauss
ians/00000000.ply
2026-01-09 16:54:34.833 | INFO     | __main__:save_gaussians_to_ply:239 - Saved transformed PLY to: /mnt/data_hdd/zhiyw/test/01pose/original/ply/00000000.ply
2026-01-09 16:54:35.081 | INFO     | __main__:main:322 -  Processed frame 00000000
Processing frames: 100%|████████████████████████████████████████████████████████████████████████████████| 1/1 [00:11<00:00, 11.50s/it]
2026-01-09 16:54:35.082 | INFO     | __main__:main:328 -  All frames processed successfully!

============================================================
TRANSFORMATION SUMMARY
============================================================
Input directory: /home/zhiyw/Desktop/AnimatableGaussians/test_results/avatarrex_zzr/avatar/thuman4__01_modified_free_view/batch_250000/pca_20_sigma_2.00/posed_gaussians
Output directory: /mnt/data_hdd/zhiyw/test/01pose/original
Frames processed: 0 to 0
Output format: ply
Transformation matrix applied:
[[ 1.417337 -3.719793  5.757977 20.600479]
 [-0.786601 -5.929179 -3.63677   2.822414]
 [ 6.80973   0.089329 -1.61852  25.250027]
 [ 0.        0.        0.        1.      ]]
============================================================
```

Step 4: 使用 python /home/zhiyw/Desktop/ml-hugs/hugs/renderer/render_sequence_firstcamera.py 渲染

```
(hugs) zhiyw@dst-jyk:~/Desktop/ml-hugs$ python hugs/renderer/render_sequence_firstcamera.py   --human_pt_dir "/mnt/data_hdd/zhiyw/test/01pose/djr/pt/"   --scene_pt "/home/zhiyw/Des
ktop/animatable_dataset/scene/djr/djr_3dgs.pt"   --output_dir "/mnt/data_hdd/zhiyw/test/01pose/image"   --start_frame 0   --end_frame  0  --camera_json "/home/zhiyw/Desktop/ml-hugs
/mip_camera/lab_train_camera_params/djr_1m.json"   --render_mode human_scene
2026-01-09 16:55:33.009 | INFO     | __main__:main:229 - Loading scene data from: /home/zhiyw/Desktop/animatable_dataset/scene/djr/djr_3dgs.pt
2026-01-09 16:55:34.101 | INFO     | __main__:main:231 - Scene loaded: 2567633 Gaussians
2026-01-09 16:55:34.101 | INFO     | __main__:main:235 - Starting batch rendering frames 0 to 0
Rendering frames:   0%|                                                                                                | 0/1 [00:00<?, ?it/s]CUDA Kernel: Fisheye GS
2026-01-09 16:55:34.598 | INFO     | __main__:save_rendered_image:179 - Frame 00000000 rendered and saved
Rendering frames: 100%|████████████████████████████████████████████████████████████████████████████████| 1/1 [00:00<00:00,  1.82it/s]
2026-01-09 16:55:34.652 | INFO     | __main__:main:306 - Completed! 1/1 frames rendered successfully.
```

生成的图片如下





情况二： 需要生成新的位置摄像头

方法一 使用四元数生成虚拟摄像头

Step1: 使用四元数生成 json 文件

1. 使用/home/zhiyw/Desktop/ml-hugs/data/neuman/dataset/lab/sparse/camera.txt 调整摄像头类型（pinhole or fisheyes）
2. 使用/home/zhiyw/Desktop/ml-hugs/data/neuman/dataset/lab/sparse/image.txt 调整摄像头位置

```
# Image list with two lines of data per image:
#   IMAGE_ID, QW, QX, QY, QZ, TX, TY, TZ, CAMERA_ID, NAME
#   POINTS2D[] as (X, Y, POINT3D_ID)
# Number of images: 40, mean observations per image: 1477.504854368932
57 0.714468 0.699646  0.002177 -0.005021  -0.224419 -0.03972 3.042475  1 00038.png
```

Qw Qx Qy Qz Tx Ty Tz

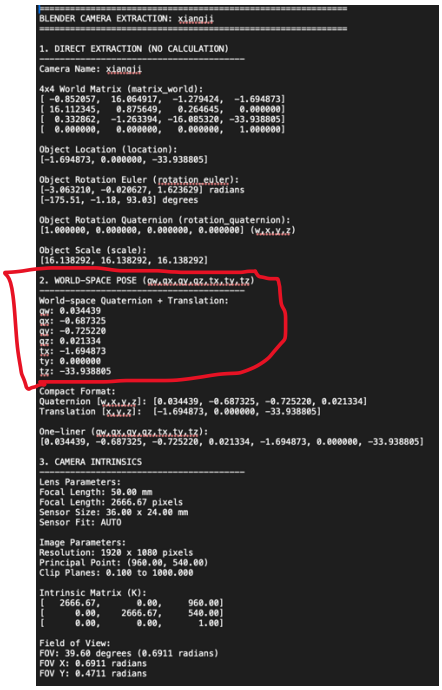
Step 2: 在 hugs 的环境中 run python run_extract_camera.py --seq lab. --split. train. --output (地址自选) 就会生成新的 json 文件

方法二 使用 blender 生成虚拟摄像头

Step1： 高斯场景导入 blender （File → Import → PLY (.ply)）

Step2： 在想要观测人物的位置创建虚拟摄像头（Shift + A → Camera）

Step3： 使用/home/zhiyw/Desktop/ml-hugs/scripts/ blender_camera.py 导出摄像头参数，导出的 txt 文件如图



使用导出后的 qw,qx,qy,qz,tx,ty,tz 来代替 image.txt 中的 qw,qx,qy,qz,tx,ty,tz

Step4 ：在 hugs 的环境中 run python run_extract_camera.py --seq lab. --split. train. --output (地址自选) 就会生成新的 json 文件

情况三 ：使用 raw data 生成 smplx 参数以适配 animatable_gaussians pipeline

我在 GitHub 提问怎样把 easymocap 得到的 smplx 参数转换成标准 smplx 参数的问题有人回答我了，具体链接在 https://github.com/lizhe00/AnimatableGaussians/issues/8#issuecomment-3650704853

Figure3：问题的回答截图

如果能通过这个方法生成标准 smplx 参数， 那么 dna rendering 就可以作为 animatable_gaussians 的数据集使用（animatable_gaussians 的 input：1.smplx 参数，2.rgb image， 3.mask， 4.camera calibration）

情况四 ：使用新的人物与场景对齐

Step1: 使用 cloudcompare 导入人物和场景

Step2: 选中人物点云，将人物点云放置在场景点云的地面上

Step3：导出 trans