ActiveMoCap: Optimized Viewpoint Selection for Active Human Motion Capture Supplementary Material

	CMU-Walk	CMU-Dance	CMU-Run	Total
Ours (Active)	0.26±0.03	0.22±0.04	0.44±0.04	0.31±0.10
Constant rot. (CW)	0.22±0.004	0.18±0.011	0.41±0.019	0.27±0.10
Constant rot. (CCW)	0.35±0.01	0.24±0.04	0.41±0.02	0.34±0.08

Table 1. **Results of drone full flight simulation**, as Table 2 of our main document, reporting the error of constant rotation in both directions. "CW" and "CCW" stand for "clock-wise" and "counter clock-wise" respectively. In general, the active trajectory's error values are in between the error values of constant rotation in the right and left directions.

1. Supplementary Video

The supplementary video provides a short overview of our work and summarizes the methodology and results. It includes video results of our active trajectories for both the teleportation and simulated flight cases.

2. Drone Flight Simulation

We mention in Section 4.1 of our main document that we obtain the constant rotation trajectory on simulated drone flight, albeit with varying rotation direction. We report the results of constant rotation in both directions in Table 1, along with our active trajectory's results. The results show that in general, the active trajectory's error values are in between the constant rotation to the right and the left. This is because the active trajectory's direction varies, however the trajectory is equivalent to constant rotation.

3. The Drone Flight Model

As we mention in Section 3.3 of our main document, in order to accurately predict where the drone will be positioned after passing it a goal velocity, we have formulated a drone flight model.

Ablation Study. We replace our drone flight model with uniform sampling around the drone. This is illustrated in Figure 1. We evaluate the performance of our active decision making policy with the uniform sampling in Table 2. The trajectories found using this sampling policy is shown in Figure 2. We find that the algorithm cannot find the constant rotation policy when we remove the drone flight model and in turn, performs worse.

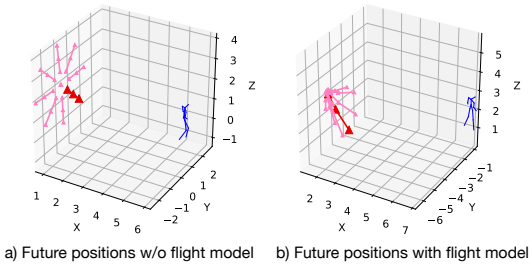

Figure 1. **The predicted future positions of the drone** (a) without using our flight model and (b) using our flight model.

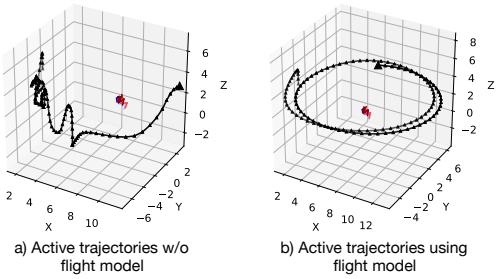

Figure 2. **The trajectories drawn by our active decision making policy** (a) without using our flight model and (b) using our flight model. We are able to find the well performing policy of constant rotation when we are using more realistic sampling of future drone positions, found using our drone flight model.

	CMU-Dribble	CMU-Sitting	CMU-Dinosour	Total
Active with Flight Model	**0.28**±0.006	0.15±0.007	**0.12**±0.02	**0.18**±0.07
Active w/o Flight Model	0.65±0.09	0.48±0.09	0.22±0.07	0.45±0.18
Constant Rot.	**0.28**±0.006	**0.14**±0.002	0.17±0.02	0.20±0.06

Table 2. **Ablation study on the importance of having a drone flight model**. We show 3D pose accuracy on simulated drone flight using noisy ground truth for estimating $\mathbf{M}$ and $\mathbf{L}$. We show that we have a large improvement when we use our flight model to predict the future locations of the drone. Using a flight model allows us to find the same trajectories as constant rotation.

4. Results with Openpose and Liftnet

We evaluate our results on the toy example case, using the networks of [2] and [4] to find the 2D pose detections $\mathbf{M}$ and 3D relative pose detections $\mathbf{L}$. The results are reported in Table 3. We outperform the baselines significantly for the real image dataset MPI-INF-3DHP. For the synthetic images, somes we are outperformed by random, but its error has much higher standard deviation and the difference between ours and random is within 1 standard deviation.

We outperform the baselines significantly in the real image dataset as compared to the synthetic datasets because the error of network [2] for real data is much lower than for synthetic data. We verify this by comparing the normalized 2D-pose estimation errors of a synthetic sequence and a sequence taken from the MPI-INF-3DHP dataset. We find that the normalized average error of [2] of the synthetic sequence is 0.10 with 0.08 standard deviation, whereas the normalized average error of the real image sequence is 0.06 with 0.06 standard deviation. Therefore, the unrealistically high noise of OpenPose on the synthetic data deprives strong conclusions from the first three columns of Table 3.

Oracle still performs very well for synthetic images in this case, but oracle makes decisions knowing the results of [2] for all candidate locations. However, this is impossible in practice due to the inherent uncertainty.

When the 2D pose detector is not unreliable, as in the case of Table 1 of our main document, we outperform random on all cases, well outside 2 standard deviations.

For the case of the MPI-INF-3DHP dataset, we remove the 4 ceiling cameras for this set of experiments. Since the networks of [2] and [4] were not trained with views from such angles they give highly noisy results which would also add noise to the values we report.

	CMU-Walk	CMU-Dance	CMU-Run	MPI-INF-3DHP.	Total
Oracle	0.13±0	0.15±0	0.16±0.0005	0.17±0.0005	0.15±0.01
Ours (Active)	**0.16**±0.005	0.25±0.0009	0.25±0.002	**0.21**±0.0008	**0.22**±0.04
Random	0.17±0.004	**0.24**±0.01	**0.24**±0.005	0.28±0.03	0.23±0.04
Constant Rot.	0.20±0.002	0.28±0.02	0.28±0.001	0.29±0.007	0.26±0.04
Constant Angle	0.71±0.50	0.76±0.37	0.69±0.22	1.26±0.53	0.72±0.03

Table 3. **3D pose accuracy on toy experiment**, using [5, 4] for estimating $\mathbf{M}$ and $\mathbf{L}$. We outperform all predefined baseline trajectories for the real image dataset, MPI-INF-3DHP. As for the cases with synthetic input, we achieve comparable results with random, albeit with much lower standard deviation.

5. Further Details About Simulation Environment

To test our algorithms we use the AirSim [3] drone simulator, a plug-in built for the Unreal game engine. An image from the simulator is shown in Figure 3.

AirSim provides a Python API which can be used to control the drone realistically, since it uses the same flight controllers as used on actual drones. The position and orientation of the drone can be retrieved from the simulator according to the world coordinate system, which takes the drone's starting point as the origin. The drone can be commanded to move to a with a specified velocity for a specified duration. We have added functionalities to the simulator to control a human character, get ground truth information about the character and animate it with motions from the CMU Graphics Lab Motion Capture Database [1].

Figure 3. **Image of the simulation environment**, AirSim.

For experiments requiring teleportation we use the simulator in "ComputerVision" mode, whereas for experiments simulating flight we use "Multirotor" mode.

References

[1] CMU Graphics Lab Motion Capture Database. `mocap.cs.cmu.edu`.

[2] Z. Cao, T. Simon, S. Wei, and Y. Sheikh. Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. In *Conference on Computer Vision and Pattern Recognition*, pages 1302–1310, 2017.

[3] S. Shah, D. Dey, C. Lovett, and A. Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017.

[4] B. Tekin, P. Marquez-Neila, M. Salzmann, and P. Fua. Learning to Fuse 2D and 3D Image Cues for Monocular Body Pose Estimation. In *International Conference on Computer Vision*, 2017.

[5] B. Zhao, X. Wu, Z.-Q. Cheng, H. Liu, and J. Feng. Multi-View Image Generation from a Single-View. In *arXiv Preprint*, 2017.