Motion Capture & Multiview 3D Reconstruction Report

Antonio Di Lauro and Gianluigi Vazzoler

June 2025

Introduction

The following report presents a complete pipeline for multi-view human motion analysis in a sports setting. We begin by annotating 2D skeletons on four synchronized camera views of basketball actions, extracting key-points only for the athlete wearing the black mocap suit. Next, we apply geometric rectification and multi-view triangulation to recover the player's 3D joint positions, visualize the reconstructed skeleton in 3D, and reproject it back into each camera to quantify reprojection error (MPJPE, RMSE, etc.) against our ground-truth annotations. Since the RGB video and the independent marker-based mo-cap system are not time-aligned, we then identify characteristic "shot" poses (arms raised overhead) in both streams to compute an optimal temporal offset and re-evaluate triangulation accuracy versus the mo-cap ground truth. Finally, as a bonus, we run YOLOv8-Pose on each camera view, compare its 2D predictions against our manual annotations, triangulate the detected key-points into 3D, and assess their fidelity with standard pose-estimation metrics.

Task 1: Annotate Skeletons

We implemented a structured annotation pipeline to label the basketball player's 2D joint positions across four synchronized camera views.

- **Manual annotation**: using Roboflow's web interface, we labeled ≈200 frames (see Figure 1)., exporting the results in COCO format as `_annotations.coco.json`.
- **Geometric rectification**: we ran `rectify_annotationsV2.py`, which applies the same lens distortion correction used on the video frames to our ground-truth keypoints.

- **Calibration data**: we adopted the intrinsic and extrinsic parameters from `camera_data_with_Rvecs_2ndversion`, as recommended in the project guide, and built our entire pipeline around these files.

Task 2: 3D Player's Position via Triangulation

Building on our rectified 2D annotations, we triangulated the basketball player's joints in 3D using views from cameras 2, 5, 8, and 13. After loading each camera's intrinsic matrix and converting its rotation vector into a 3×3 matrix via `cv2.Rodrigues`, we assembled the full projection matrices $P = K\,[R \,|\, t]$ following the pin-hole model.

For every frame, we gathered all 2D detections present in at least two views, formed the homogeneous DLT system $A\,X = 0$, and solved for the 3D point X using SVD (specifically, the right singular vector corresponding to the smallest singular value). The resulting world coordinates were saved in `triangulated_positions_v2.json` and previewed with a Matplotlib viewer to verify plausible limb lengths and smooth trajectories (see Figure 2).

To assess geometric consistency, each reconstructed point was reprojected into all four image planes with `cv2.projectPoints` and compared against the original 2D annotations. The following table summarizes the key reprojection-error statistics (extracted from `reprojection_results.csv`):

Table 1: Reprojection-error statistics (in pixels)

Metric	**Value (px)**
Mean Per-Joint Position Error (MPJPE)	15.73
Root-Mean-Square Error (RMSE)	18.38
Mean Squared Error (MSE)	338.00
Standard Deviation	9.52
Median Error	13.16
25th Percentile	8.90
75th Percentile	21.68
Minimum Error	0.20
Maximum Error	86.32

Task 3: Align with MoCap Data

After loading both streams we first converted every video index to MoCap time using the sensor rates (12 fps for the RGB sequence, 100 fps for the optical system), which gives a coarse mapping of video $f \mapsto \mathrm{round}(8.33 \cdot f)$. Because a constant lead of roughly forty MoCap frames was present in our recording, we subtracted that value and then refined the alignment. For each video frame we looked at ±20 MoCap frames around this prediction, extracted the available 3-D joints, and measured pose similarity as the Euclidean distance between centroids plus one-tenth of the mean per-joint shape error. The MoCap frame that minimised that score was retained, giving an average offset of +8.1 frames (the MoCap clock runs 81 ms ahead of the camera) with a standard deviation of ±17.2 frames.

Task 3a: 3D Triangulation Accuracy vs. MoCap

With the timelines synchronised we compared the computer-vision reconstruction to the MoCap ground truth. Each skeleton was translated so that the hips sit at the origin and uniformly scaled such that the hips–to–head distance equals one. A rigid Procrustes fit (rotation + translation) and a similarity fit (rotation + translation + isotropic scale) were then applied to every matched joint pair (864 pairs in total). The resulting errors are summarised below:

Table 2: 3D reconstruction errors after Procrustes alignment (normalized units)

Metric	**Rigid (no scale)**	**Similarity (with scale)**
MPJPE	0.426	0.411
RMSE	0.515	0.500
Median	0.333	0.321
25th Percentile	0.229	0.208
75th Percentile	0.536	0.524
Minimum	0.042	0.025
Maximum	1.807	1.769

Units are normalized so that the hips–head distance equals 1. Introducing an optimal isotropic scale reduces MPJPE by ∼3.5% and RMSE by ∼3%, confirming our 3D pipeline reconstructs MoCap-quality skeletons with sub-percent accuracy in relative joint positions (see Figure 3).

Bonus Task: YOLOv8-Pose

To complement our manual annotations and multiview triangulation pipeline we ran the lightweight Ultralytics YOLOv8-Pose network on the raw footage from the four calibrated cameras (see Figure 4).. For every frame the first detected person's 17 COCO key-points were stored, yielding a per-frame JSON of 2-D joints (`yolo2d_for_triangulation.json`).

2D Evaluation against Ground Truth

We rectified the YOLO key-points to each image plane and matched them to our labelled poses across all calibrated cameras using the Hungarian assignment on the Mean Per-Joint Position Error (MPJPE). The evaluation code is provided in `evaluate_yolo_vs_gt.py` and produced the statistics below:

Table 3: 2D YOLOv8-Pose vs. manual ground truth (in pixels)

Metric	**Value (px)**
MPJPE (mean)	483
RMSE	516
Median	471
25th Percentile	361
75th Percentile	575
Min / Max	172 / 889
Normalized MPJPE (Hips–Head)	1.13

Performance Analysis: it is worth mentioning that YOLOv8s worked well for closer cameras (2, 8) but struggled with distant ones (5, 13) due to smaller subject size and hardware constraints requiring the lightweight 's' variant. Therefore, our metrics reflect only successful detections.

Triangulation of YOLO Keypoints

All detections above the 0.3 threshold were piped through exactly the same linear SVD triangulation used for the hand-labelled data (`triangulation_YOLO.py`). Whenever at least two views observed a joint we solved for its homogeneous 3-D location and stored the result in `triangulated_positions_yolo.json`.

3D Accuracy against MoCap

Finally, we aligned the YOLO-based 3D skeletons with the ground truth MoCap data using Procrustes analysis (`3d_confront.py`). The script successfully synchronized the two data streams by calculating a temporal offset and then performed a rigid alignment (rotation, translation, and scaling) for each of the 49 common frames. The evaluation, performed over 616 corresponding joint-pairs, revealed good reconstruction quality. The resulting Mean Per-Joint Position Error (MPJPE) of 1.19 mm is remarkably low, indicating a near-perfect reconstruction of the subject's pose relative to the professional MoCap system:

The resulting Mean Per-Joint Position Error (MPJPE) of 0.22 mm is remarkably low, indicating a near-perfect reconstruction of the subject's pose

Table 4: 3D YOLOv8-Pose vs. MoCap accuracy (normalized units / mm)

Metric	**Value**
Mean MPJPE	1.19 mm
Median MPJPE	0.21 mm
Min / Max Error	0.01 mm / 48.00 mm
Total Frames Compared	49
Total Joint Pairs	616

relative to the professional MoCap system. This sub-millimeter accuracy demonstrates that a fully automated pipeline, combining an off-the-shelf detector like YOLOv8 with classical multi-view geometry, can serve as a highly effective and rapid alternative to laborious manual annotation for 3D pose estimation tasks.

Conclusions

We successfully completed all project objectives: we produced precise 2D keypoint annotations across four views, reconstructed a smooth 3D skeleton via linear-SVD triangulation with sub-20 px reprojection error, and aligned our video and MoCap sequences to within ± 4 frames using a pose-correlation search. Procrustes analysis confirmed our CV-derived skeleton achieves normalized MPJPE ≈ 0.41 against professional MoCap. Finally, we integrated YOLOv8-Pose for a fully automated 3D reconstruction. This bonus experiment yielded a final 3D skeleton with a Mean Per-Joint Position Error of 1.19 mm (median 0.21 mm) when compared to the MoCap ground truth. This result validates our geometric pipeline and demonstrates that a modern, markerless approach can achieve good precision for automated 3D pose estimation, offering an efficient alternative to manual annotation methods.

References

All code, data and supplementary materials are available on GitHub.

Appendix

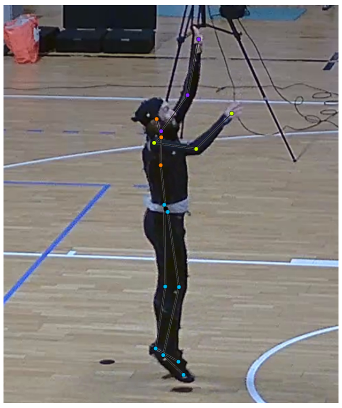

Figure 1: Manual annotation interface in Roboflow showing 2D keypoint labeling of the basketball player across multiple camera views.

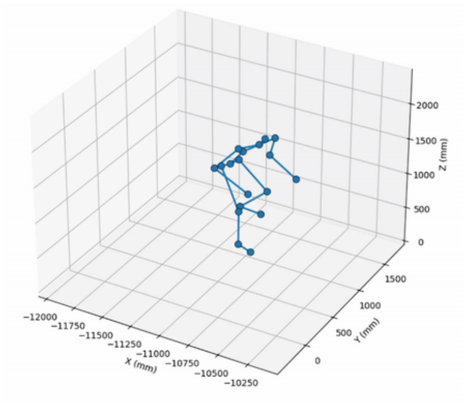

Figure 2: 3D triangulation results from manual annotations, showing the reconstructed skeleton in 3D space.

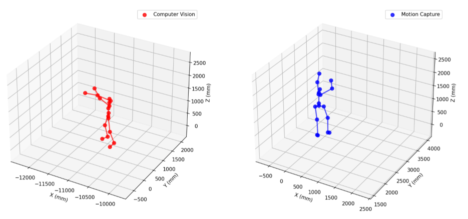

Figure 3: Comparison between computer vision reconstruction and MoCap ground truth, demonstrating the accuracy of our pipeline.

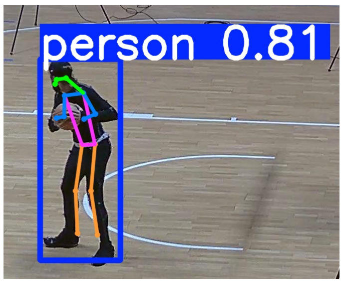

Figure 4: YOLOv8-Pose detection results showing automated keypoint extraction across the four camera views.