A Real-Time Pedestrian Top-view Tracking System for Multicamera Environment

Jihoon Choi
Department of Computer Science
Kyungpook National University
Daegu, South Korea
nobless@knu.ac.kr

Young-Woo Kwon
Department of Computer Science
Kyungpook National University
Daegu, South Korea
ywkwon@knu.ac.kr

***Abstract*—This paper presents a practical and deployable system for estimating the planar positions of pedestrians using existing monocular CCTV infrastructure, without requiring depth sensors or additional hardware. Our approach leverages a modern object detector (YOLOv8) to obtain pedestrian bounding boxes from camera streams. For each detection, we approximate the footpoint (i.e., the contact point with the ground) and transform its image coordinates to real-world ground-plane coordinates using a pre-calibrated homography matrix. This yields a Top-view localization of each individual. The system is designed to aggregate data from multiple camera streams, sending the localized positions to a front-end application that visualizes all pedestrians on a unified control map in real-time. This low-cost solution enables wide-area situational awareness, making it suitable for applications in smart city management, security surveillance, and digital twin environments.**

***Index Terms*—Pedestrian Localization, Monocular Camera, Homography, Top-view, CCTV Surveillance**

I. Introduction

Estimating the real-world location of pedestrians is fundamental for a wide range of applications, including urban analytics, safety monitoring, and crowd management [1] [2]. While solutions using LiDAR, stereo cameras, or depth sensors provide high accuracy, their deployment is often limited by high cost and the need for specialized hardware. In contrast, vast networks of monocular CCTV cameras are already installed in public and private spaces. This raises a critical question: Can we achieve reliable ground-plane localization using only this existing infrastructure?

Approaches using RGB-D cameras [3] [4] can improve tracking accuracy with depth information. However, these solutions face significant limitations. The active infrared sensors in many RGB-D cameras are unreliable in outdoor settings due to sunlight interference and have a limited effective range. The quality of the depth data can also be degraded by material properties, such as shiny or dark surfaces. Consequently, the need for specialized hardware and these operational constraints make RGB-D cameras unsuitable for a universally deployable solution that relies on existing infrastructure.

This paper proposes an end-to-end system that answers this question affirmatively for scenes with approximately planar ground. We present a practical and low-cost system that leverages existing CCTV cameras to detect and localize pedestrians using a bounding object detection box [5] in real-time. The core of our system consists of three main stages: (1) real-time pedestrian detection and tracking using YOLOv8, (2) footpoint extraction from pedestrian bounding boxes, and (3) coordinate transformation from the 2D image plane to a 2D world ground plane using homography [6]. This approach provides location information without complex camera calibration procedures, making it easily deployable.

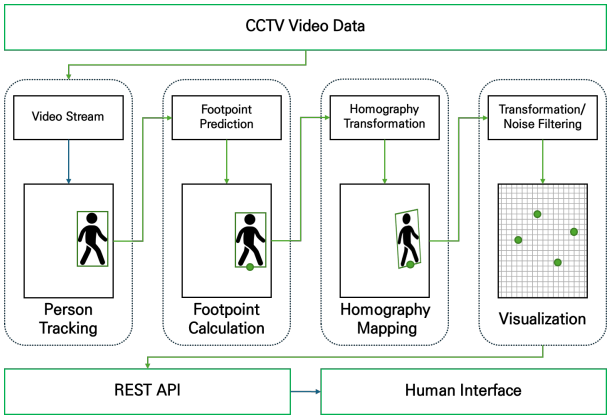

Fig. 1. System Architecture. Each CCTV stream is processed to extract Top-view coordinates, which are then aggregated and visualized on a unified map in the front-end.

II. System Architecture and Method

Our proposed system consists of a backend processing pipeline for each camera and a frontend for centralized visualization. The overall system architecture is illustrated in Fig. 1. The backend processes CCTV video streams to detect pedestrians, extract their footpoints in the image plane, and transform these coordinates into Top-view world coordinates. The resulting data is then sent to a front-end application for real-time visualization on a unified map.

A. Object Detection and Tracking

For real-time pedestrian detection and tracking, we use YOLOv8 [7], the latest version of the You Only Look Once (YOLO) family of models. YOLOv8 provides high accuracy and fast inference speeds, making it suitable for real-time applications. Its built-in tracking capabilities allow it to assign a

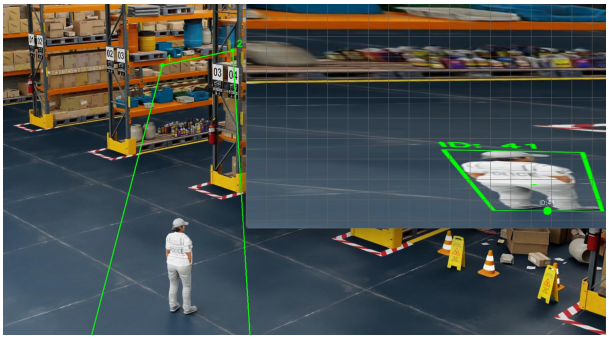

Fig. 2. Visualization of the backend Yolo and homography transformation processing.

unique ID to each pedestrian and track them across consecutive frames without requiring an external tracking algorithm like DeepSORT [8]. For each detected pedestrian in a video frame, the model outputs a bounding box defined by its coordinates $(x_{min}, y_{min}, x_{max}, y_{max})$ and a tracking ID.

B. Footpoint Extraction

CCTV video was generated using Isaac Sim for this study. The pedestrian's footpoint was calculated as the bottom center of the bounding box. In the image coordinate system, the footpoint P_{img} is defined as follows:

$$P_{img} = \left(\frac{x_{min} + x_{max}}{2}, y_{max} \right) \tag{1}$$

This method is computationally efficient and independent of the specific object detector used.

C. Top-view Transformation via Homography

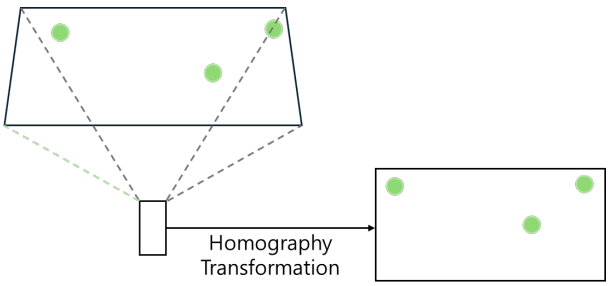

Fig. 3. Homography transformation from image coordinates (left) to Top-view coordinates (right) using four user-defined correspondence points.

A homography is a projective transformation that maps points from one plane to another [9]. In our case, we use a 3×3 homography matrix, H, to map points from the 2D image plane to the 2D ground plane in the real world. Given a foot point p_{img} in homogeneous coordinates $\tilde{p}_{img}$, its corresponding world coordinate p_g on the ground plane is computed as:

$$\tilde{p}_g = H \cdot \tilde{p}_{img}$$

The resulting homogeneous coordinate $\tilde{p}_g = [x'_g, y'_g, w'_g]^\top$ is converted back to Cartesian coordinates $[x_g, y_g]^\top$ by dividing by w'_g. The matrix

D. Implementation and Visualization

The proposed system was implemented using a client-server architecture. The backend is developed in Python with OpenCV [10] for image processing and homography calculations, while the frontend is a web application built with React [11]. Communication between the backend and frontend is handled in real-time using WebSockets [12].

Users interact with the system via the web UI. They can upload a video source or provide a stream URL. Then, using mouse clicks, they select four points on the video frame to define the ROI for the homography transformation. The backend receives the video and ROI coordinates, computes the homography matrix, and begins processing the video stream.

For each frame, the backend detects and tracks pedestrians, extracts their footpoints, and transforms them into Top-view coordinates. These coordinates, along with their tracking IDs, are sent to the frontend in real-time. The frontend provides two views: (1) the original video stream with bounding boxes and tracking IDs overlaid, and (2) an integrated Top-view control map where each pedestrian is represented by a dot and their ID, as shown in Fig. 4. This dual visualization allows operators to intuitively understand crowd density, movement paths, and individual trajectories.

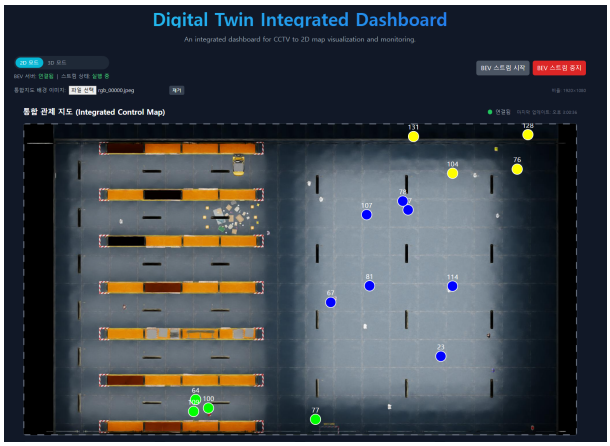

Fig. 4. Example of the integrated control map GUI. The right panel shows the Top-view map with real-time positions of pedestrians, while the left panels show the source camera feeds.

III. Kalman Filter for Tracking Stabilization

We optionally apply a constant-velocity 2D Kalman filter to stabilize the time series of pedestrian locations on the Top-view plane. The filter reduces detection jitter via short-horizon prediction and noise suppression and improves positional reliability during brief occlusions.

This formulation follows standard Kalman filter theory and practice [13], [14] and reflects common usage in vision-based pose/trajectory estimation [15].

Part of our evaluation uses synthetic CCTV data generated with NVIDIA Isaac Sim. We record the world-frame coordinates of simulated human agents at every frame to obtain ground-truth pedestrian positions. These ground-truth tracks

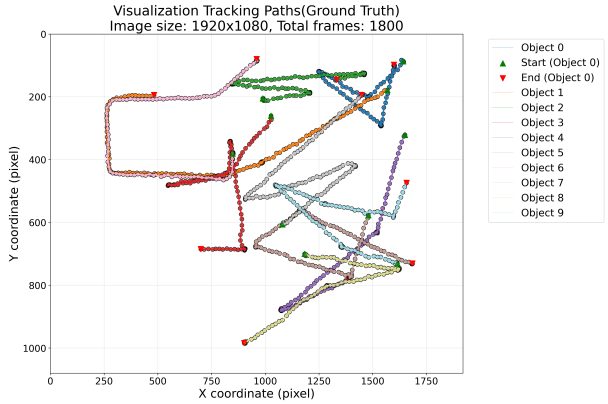

Fig. 5. Ground-truth pedestrian positions in a CCTV scenario generated with NVIDIA Isaac Sim.

are used to visualize interpolated trajectories and to support both quantitative and qualitative evaluation. Figure 5 illustrates the distribution of true pedestrian positions in the same scene.

A. State-space model

We define the state and measurement vectors as $\mathbf{x} = [x,\ y,\ v_x,\ v_y]^\top$ and $\mathbf{z} = [x,\ y]^\top$, respectively. For a time step of Δt, the linear motion and measurement models are

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

We use simplified covariances $\mathbf{Q} = q\,\mathbf{I}_4$ and $\mathbf{R} = r\,\mathbf{I}_2$. The prediction and update follow the standard Kalman filter:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}\,\hat{\mathbf{x}}_{k-1|k-1} \tag{2}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}\,\mathbf{P}_{k-1|k-1}\,\mathbf{F}^\top + \mathbf{Q} \tag{3}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\,\mathbf{H}^\top \left(\mathbf{H}\,\mathbf{P}_{k|k-1}\,\mathbf{H}^\top + \mathbf{R}\right)^{-1} \tag{4}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}\,\hat{\mathbf{x}}_{k|k-1}\right) \tag{5}$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\,\mathbf{P}_{k|k-1} \tag{6}$$

B. Integration in the Top-view pipeline

For each tracking ID, we maintain an independent filter instance. The bottom-center footpoint of the bounding box is first mapped to Top-view coordinates $(x^{Top-view}, y^{Top-view})$ via homography and then used as the measurement for the per-frame update. The initial state is set with the first observation and zero velocity. When a track disappears, its filter is discarded.

The filter is integrated into the backend's Top-view overlay stage and can be toggled on/off. The main hyperparameters are process noise q, measurement noise r, and the sampling interval Δt. Larger r increases smoothing by trusting measurements less, while larger q improves responsiveness to acceleration but may pass more noise. Δt should match the actual frame interval.

C. Practical settings and observations

We evaluated multiple $(r, q, \Delta t)$ combinations to balance jitter reduction and trajectory continuity. Small r yields high responsiveness but more jitter; large r reduces jitter but degrades responsiveness to abrupt direction changes. Increasing q helps follow sharp turns but can transmit noise if too large. In practice, we recommend moderate r, low-to-moderate q, and Δt aligned with the video frame rate for a good real-time stability trade-off.

D. Evaluation protocol and results

To quantify temporal stability and smoothness, we evaluate tracks stored in per-session databases using four metrics computed per track and averaged across tracks: (1) jitter (standard deviation of per-frame displacement), (2) curvature (mean absolute change in direction in radians), (3) gap rate (fraction of missing frames over track lifespan), and (4) a composite stability score in $[0, 100]$ defined as

$$\text{score} = 100 \cdot \left(1 - [0.5\,\tilde{j} + 0.3\,\tilde{c} + 0.2\,\tilde{g}]\right),$$

where $\tilde{j}, \tilde{c}, \tilde{g}$ are robustly min–max normalized via the 5th and 95th percentiles within a session. This protocol is implemented in our analysis script.

The test data used in our experiments was generated in a virtual environment using NVIDIA Isaac Sim, simulating arbitrary CCTV surveillance scenarios. We created synthetic videos featuring pedestrians moving in various locations and adjusted different filter parameters based on these simulated scenes to assess stability (e.g., jitter, curvature, etc.).

We conducted two ablations with $\Delta t = 1$:

TABLE I
PER-SESSION AVERAGES ANALYSIS FIXED $r = 4$.

Session	Stability ↑	Jitter ↓	Curv. (rad) ↓	Gap rate ↓
OFF	66.20	1.935	1.415	0.0079
$q = 1,\ r = 4$	66.77	1.666	1.213	0.0088
$q = 2,\ r = 4$	68.02	1.756	1.249	0.0079
$q = 4,\ r = 4$	66.42	1.762	1.262	0.0086
$q = 6,\ r = 4$	67.86	1.786	1.265	0.0086

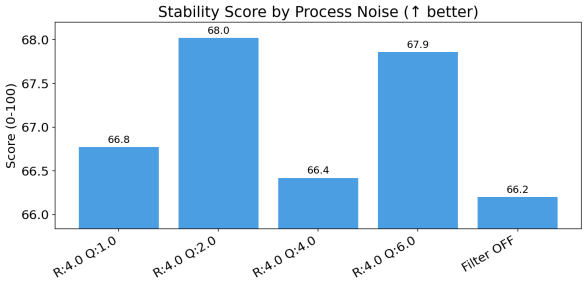

Fig. 6. Stability score comparison across different process noise q (responsiveness vs. smoothness trade-off).

- Varying process noise q with fixed $r = 4$. The best composite score was obtained at $q = 2$ with a stability score of ≈ 68.0, improving over Filter OFF (≈ 66.2) by about $+1.8$.

TABLE II
PER-SESSION AVERAGES ANALYSIS FIXED $q = 2$

Session	Stability ↑	Jitter ↓	Curv. (rad) ↓	Gap rate ↓
OFF	66.20	1.935	1.415	0.0079
$r = 0.5,\ q = 2$	68.64	1.860	1.296	0.0076
$r = 1.0,\ q = 2$	66.64	1.811	1.274	0.0085
$r = 2.0,\ q = 2$	67.29	1.759	1.266	0.0087
$r = 4.0,\ q = 2$	68.02	1.756	1.249	0.0079

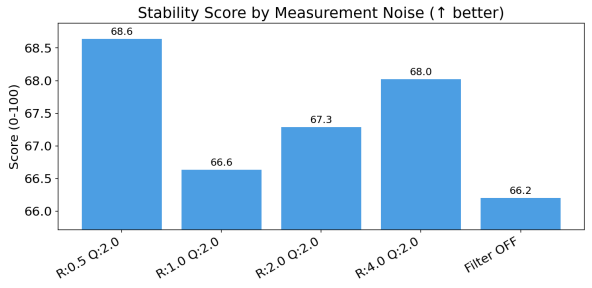

Fig. 7. Stability score comparison across different measurement noise r (lower jitter/curvature yield higher scores).

- Varying measurement noise r with fixed $q = 2$. The best setting was $r = 0.5$ with a stability score of ≈ 68.6 (vs. OFF ≈ 66.2, $+\approx 2.4$); $r = 4.0$ was the next best at ≈ 68.0, followed by $r = 2.0$ and $r = 1.0$.

Overall, moderate measurement noise ($r \approx 0.5$) and low-to-moderate process noise ($q \approx 2.0$) yielded the best stability in our data, corroborating the practical guidance above.

E. Ablation option

Depending on deployment needs, the Kalman filter can be disabled to use raw Top-view coordinates as is. This is useful when assessing interactions with multi-camera fusion or downstream smoothing methods.

IV. CONCLUSION AND FUTURE WORK

In this paper, we presented a practical and cost-effective system for real-time pedestrian localization that leverages existing CCTV infrastructure. Our approach combines the high performance of the YOLOv8 object detector, a simple but effective footpoint approximation, a user-calibrated homography transformation, and a lightweight 2D Kalman filter for per-track temporal smoothing. This enables robust conversion of multi-camera 2D streams into a unified Top-view pedestrian map with enhanced stability. The overall pipeline is easily deployable and scalable, making it applicable to a wide range of domains, including security surveillance and urban analytics.

The integration of the 2D Kalman filter in our pipeline brings several practical advantages. By predicting and smoothing the per-person footpoint positions frame by frame, the Kalman filter significantly reduces visual jitter and discontinuities in the reconstructed trajectories. Our results show that appropriate tuning of filter parameters (process and measurement noise, sampling interval) provides a strong trade-off between responsiveness to real motion changes and temporal stability—essential for both human visualization and downstream applications. Moreover, the filter's modular design allows it to be toggled or replaced depending on deployment constraints and the presence of more complex smoothing or multi-sensor fusion modules.

Looking ahead, several research directions could further enhance our work:

- **Advanced Kalman Filter Extensions:** Exploring more sophisticated state models (e.g., constant acceleration, maneuvering models), adaptive noise estimation, or switching filters could provide improved accuracy and robustness, especially in highly dynamic scenes or with irregular frame intervals.
- **Improved Re-Identification (Re-ID):** Integrating robust Re-ID models may enable consistent tracking of individuals across non-overlapping camera views, addressing one of the major challenges in multi-camera systems [16].
- **Automated Calibration:** Developing methods for automatic or semi-automatic homography calibration—such as exploiting scene geometry or vanishing lines—would significantly reduce setup labor and make deployments even more accessible [17].
- **Handling Non-Planar Scenes:** Since our method currently assumes a flat ground plane, future work could extend it to handle non-planar geometry (e.g., slopes, stairs) by incorporating 3D priors or online estimation of height variations [18].
- **Occlusion Handling:** In crowded scenes, occlusions are a persistent challenge. Fusing data from multiple overlapping camera views in combination with filtering may better resolve ambiguities and maintain robust trajectories [19] [20].

In summary, the addition of the Kalman filter not only improves the practical usability of our Top-view tracking but also provides a flexible foundation for further research on robust, automated, and scalable multi-camera pedestrian analysis.

ACKNOWLEDGEMENT

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2021-NR060080).

REFERENCES

[1] M. Singh, E. Fuenmayor, E. P. Hinchy, Y. Qiao, N. Murray, and D. Devine, "Digital twin: Origin to future," *Applied System Innovation*, vol. 4, no. 2, 2021. [Online]. Available: https://www.mdpi.com/2571-5577/4/2/36

[2] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, "Multiple object tracking: A literature review," *Artificial Intelligence*, vol. 293, p. 103448, 2021.

[3] L. Zhang, J. Sturm, D. Cremers, and D. Lee, "Real-time human motion tracking using multiple depth cameras," in *IROS*, 2012, pp. 2389–2395. [Online]. Available: https://doi.org/10.1109/IROS.2012.6385968

[4] D. Liciotti, M. Paolanti, E. Frontoni, and P. Zingaretti, "People detection and tracking from an rgb-d camera in top-view configuration: review of challenges and applications," in *International Conference on Image Analysis and Processing*. Springer, 2017, pp. 207–218.

[5] T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object detection using yolo: challenges, architectural successors, datasets and applications," *multimedia Tools and Applications*, vol. 82, no. 6, pp. 9243–9275, 2023.

[6] E. Malis and M. Vargas, "Deeper understanding of the homography decomposition for vision-based control," Ph.D. dissertation, Inria, 2007.

[7] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," https://github.com/ultralytics/ultralytics, 2023, gitHub Repository.

[8] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 3645–3649.

[9] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[10] G. Bradski, "The opencv library." *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000.

[11] Facebook Inc., "React: A JavaScript library for building user interfaces," https://react.dev, 2023, accessed: 2024-10-27.

[12] I. Fette and A. Melnikov, "The WebSocket Protocol," IETF, RFC 6455, 2011.

[13] G. Welch and G. Bishop, "An introduction to the kalman filter," University of North Carolina at Chapel Hill, Department of Computer Science, Tech. Rep., 1995.

[14] Q. Li, R. Li, K. Ji, and W. Dai, "Kalman filter and its application," in *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, 2015, pp. 74–77.

[15] F. Janabi-Sharifi and M. Marey, "A kalman-filter-based method for pose estimation in visual servoing," *IEEE Trans. Robot.*, vol. 26, no. 5, pp. 939–947, 2010.

[16] L. Song, X. Jin, J. Han, and J. Yao, "Pedestrian re-identification algorithm based on unmanned aerial vehicle imagery," *Applied Sciences*, vol. 15, no. 3, 2025. [Online]. Available: https://www.mdpi.com/2076-3417/15/3/1256

[17] L. Teixeira, F. Maffra, and A. Badii, "Scene understanding for auto-calibration of surveillance cameras," in *International Symposium on Visual Computing*. Springer, 2014, pp. 671–682.

[18] P. Xiao, F. Yan, J. Chi, and Z. Wang, "Real-time 3d pedestrian tracking with monocular camera," *Wireless Communications and Mobile Computing*, vol. 2022, no. 1, p. 7437289, 2022.

[19] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, "Multiple object tracking: A literature review," *Artificial Intelligence*, vol. 293, 05 2017.

[20] D. Stadler and J. Beyerer, "Improving multiple pedestrian tracking by track management and occlusion handling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 10 958–10 967.