3D Trajectory Reconstruction of Dynamic Objects in Digital Twins from Monocular Video

Bogwan Kim[1], Haeseong Lee[1] and Myungho Lee[1,*]

[1]Pusan National University, 2, Busandaehak-ro 63beon-gil, Geumjeong-gu, Busan, South Korea

Abstract

The growing demand for remote monitoring through digital twins highlights the importance of integrating both structural accuracy and dynamic awareness of physical spaces. While 3D reconstruction technologies enable highly precise digital twin environments, they typically remain static, failing to reflect real-time changes. Conversely, CCTV systems provide live monitoring but only as separate 2D video streams, requiring users to mentally map them to the reconstructed 3D environment. To address this gap, we propose a 2D–3D projection-based pipeline that incorporates dynamic object trajectories from monocular video into a 3D reconstructed digital twin. Our method leverages widely available indoor CCTV feeds, combining them with reconstructed static scenes and camera pose information to back-project object masks and recover placement and orientation. A stabilization filter further ensures robustness against noise and mask deformation. This approach offers a practical foundation for integrating dynamic objects into digital twins, facilitating more consistent spatial perception and real-time monitoring of remote environments.

Keywords

Digital Twin, Dynamic Object Trajectory Reconstruction, Pose Estimation, Video Surveillance

1. Introduction

Digital Twin (DT) technology is gaining significant attention as an innovative paradigm that connects the physical and digital worlds, enabling the continuous reflection of a real environment's state, behavior, and changes over time in a virtual environment [1, 2]. Unlike traditional modeling approaches, which were often limited to static representations or simplified simulations, DTs integrate heterogeneous data sources—such as sensor data, image data, and simulation results—to provide a continuously updated virtual environment [3]. This characteristic is particularly crucial in various application domains such as smart manufacturing, healthcare, urban infrastructure management, and autonomous driving, where the demand for real-time monitoring, predictive analytics, and decision support is rapidly increasing [4, 5]. In this context, the usability and reliability of a DT are directly determined by the level of fidelity with which the virtual model reflects the structural, spatial, and temporal characteristics of the physical environment [1, 6]. Therefore, fidelity has become a core concept in DT research, extending beyond mere geometric representation or physical model accuracy to a comprehensive discussion that includes the realism of dynamic interactions and behavioral patterns [2, 7].

While fidelity can be defined in various ways [4], it essentially refers to how accurately a DT captures not only the static properties of a real environment but also its dynamic states and transitions over time. For example, if a DT of a manufacturing site only reproduces the geometric shape of machinery and fails to reflect dynamic elements such as trajectories, its utility for predictive maintenance is limited [8]. Similarly, if a smart city's DT includes only static infrastructure like buildings and roads but fails to track the movement of mobile objects such as vehicles and pedestrians, it cannot sufficiently contribute to traffic flow analysis or safety decision support [9]. These examples illustrate that the value of a DT lies not merely in creating a visually precise digital replica, but in ensuring a functionally equivalent level to reality by securing spatiotemporal consistency between the physical and virtual environments [1]. However, achieving such high fidelity entails several challenges. While low-fidelity models can

APMAR'25: The 17th Asia-Pacific Workshop on Mixed and Augmented Reality, Sep. 26-27, 2025, Busan, South Korea

*Corresponding author.

kyj12348@pusan.ac.kr (B. Kim); heaseong@pusan.ac.kr (H. Lee); myungho.lee@pusan.ac.kr (M. Lee)

0009-0005-2597-4471 (B. Kim); 0009-0004-4601-937X (H. Lee); 0000-0002-0766-6484 (M. Lee)

reduce computational resource consumption, discrepancies with reality may lead to degraded prediction performance or erroneous judgments. Conversely, high-fidelity DTs require precise 3D reconstruction, robust pose estimation, and reliable dynamic object tracking, thus demanding massive computational loads and significant algorithmic complexity [10, 11]. Therefore, determining how to define and balance the level of fidelity has emerged as a key challenge in DT research, especially in application contexts that simultaneously require dynamic object recognition, real-time localization, and temporal consistency [12].

In this context, research aimed at virtually reproducing real-world scenes has continued steadily. 3D reconstruction is a prime example. Traditional pipelines for reconstructing 3D scenes from multi-view cameras have widely used Structure-from-Motion(SfM) [13] to estimate camera poses and sparse points, followed by Multi-View Stereo (MVS) [14] to produce dense depth and meshes. More recently, rapid advances in neural reconstruction methods—most notably Neural Radiance Fields (NeRF)—have made it possible to create and update high-precision 3D models of large-scale scenes [15, 16]. In particular, as the accuracy of visual localization and the conversion pipelines between mesh-based and pointcloud-based representations have matured [17, 18], it has become feasible to stably perform reconstruction and maintenance of a scene's geometry and material properties at industrial scale. These technical foundations provide the continuously high-quality updatable spatial models required by DTs.

At the same time, progress in computer vision—object segmentation [19, 20], Multi-Object Tracking(MOT) [21], and Human Activity Recognition (HAR) [22]—has made it possible to quantitatively characterize and measure object states and scene events from images and video streams. In addition, the advent of vision-language models(VLMs) supports query-centric recognition and relational/descriptive reasoning even for object and behavior categories that are not predefined, enabling robust integration of domain-specific knowledge into vision pipelines tailored to each use case [23]. These models move beyond mere visualization of static scenes in Digital Twins to enable tracking, explanation, and prediction of dynamic states.

Together, advances in 3D reconstruction and object understanding now make it feasible to operate CCTV-equipped indoor environments (e.g., manufacturing facilities) as digital twins synchronized with their physical counterparts. By leveraging a predefined 3D scene model and visual localization, segmented objects from video streams can be registered into the 3D scene, ensuring spatiotemporal consistency. In this paper, we focus on synchronizing dynamic objects and propose a method to reconstruct their motion frame by frame within a virtual environment. We assume that a reconstructed static scene, 3D mesh models of dynamic objects, and accurate camera poses are available—assumptions that align with the current state of 3D reconstruction, modeling, and localization technologies. From the input image sequences, we extract object masks and incorporate predefined object and spatial information to reinforce consistency between physical and virtual spaces, enabling high-fidelity representations of dynamic objects in DT environments. This approach provides foundational techniques for implementing dynamic digital twins in domains with frequent motion, such as manufacturing facilities and urban settings.

2. Methodology

This section introduces a pipeline for high-fidelity DT representation of dynamic objects in indoor scenes recorded by a static camera (e.g., CCTV). To this end, we assume the following are given: (i) a 3D reconstructed mesh of the static scene, (ii) a 3D mesh of the dynamic objects, and (iii) intrinsic and extrinsic parameters of the camera. In particular, the camera pose estimated within the DT is assumed to be aligned—via visual localization—with the coordinate frame of the physical camera used for capture.

Since the dynamic objects are predefined, we prompt SAM2 once at initialization to obtain per-frame masks. From the pixel distribution in each mask, we compute a principal ray, which is then projected into the world coordinate system using the camera parameters. The intersection of this ray with the object's mid-height plane yields the per-frame position, while the displacement between successive positions determines the rotation, primarily yaw. To reduce inter-frame rotational instability caused

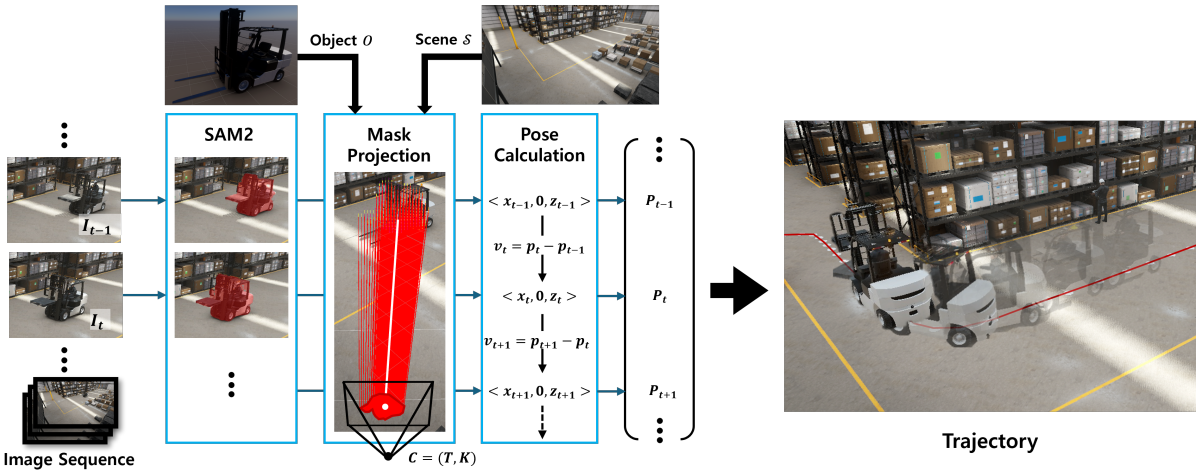

Figure 1: Overview of the proposed pipeline. Object masks are extracted from the input image sequence using SAM2. In the mask projection step, a principal ray (white line) is computed. The object's position is determined from the intersection of this ray with the mid-height plane, while rotation is estimated from positional changes across frames. A stabilization filter is then applied, and the pose list is generated with both position and rotation values. If mask loss occurs between frames, any pose values that were not calculated are interpolated.

by mask deformation and noise, we apply a stabilization filter. Finally, if the position and rotation values were not calculated due to complete mask loss, we interpolate them to maintain consistency. An overview of the pipeline is shown in Figure 1.

2.1. Problem Statement

We define the proposed algorithm as $\mathbf{F}$, as shown in Eq. 1. Here, $\mathcal{S}$ denotes the 3D scene mesh and O represents the target object model. The camera is defined as $C = (T, K)$, where $T = [T_R, T_t]$ is the 3D pose of C—with T_t representing translation and T_R representing rotation—and K denotes the intrinsic parameters. $\mathcal{I}$ denotes the monocular RGB image sequence (i.e., video) captured by C, and I_t refers to the image at frame t.

$$\mathbf{F}(\mathcal{S}, O, C, \mathcal{I}) = \mathcal{P} = \{P_t\}_{t=1}^{N} \tag{1}$$

The 3D pose of O at time t is represented as $\{x_t, y_t, z_t, R\}$, where $R \in SO(3)$. The pose of O on the ground plane of the scene model $\mathcal{S}$ at time t, computed by $\mathbf{F}$ and denoted as P_t, is defined in Eq. 2, where θ denotes the yaw angle.

$$P_t = \{x_t, 0, z_t, \theta_t\} \tag{2}$$

2.1.1. 3D Mask Projection

To project O from a 2D image onto the 3D scene, we first generate the target object mask M_t on the image I_t using SAM2. M_t is represented as an array of 2D pixels $p_i = (u_i, v_i)$. For each pixel in the mask, we define a ray $r_p(k)$ using C, as shown in Eq. 3, where k denotes the depth of ray. Finally, we compute the ray set $\mathbf{r}_t = \{r_{p_1}^{M_t}, r_{p_2}^{M_t}, ...\}$ for 3D projection.

$$r_i^{M_t}(k) = -T_R^\top T_t \; + \; k\, T_R^\top K^{-1} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}, \qquad k > 0 \tag{3}$$

However, these rays may be affected by mask noise or camera pose errors. To ensure robustness, we compute the unit vector $\hat{\mathbf{d}}$ of ray, as defined in Eq.4.

$$\mathbf{d}_r = T_R^{\top} K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}, \qquad \hat{\mathbf{d}}_r = \frac{\mathbf{d}_r}{||\mathbf{d}_r||} \tag{4}$$

The principal ray $\bar{r}^{M_t}(k)$ is then defined as the mean of the unit vectors in $\mathbf{r}_t$, as given in Eq. 5, where $|\mathbf{r}|$ denotes the size of array.

$$\bar{r}^{M_t}(k) = \frac{\sum_r^{\mathbf{r}^{M_t}} \hat{\mathbf{d}}_r}{|\mathbf{r}^{M_t}|} k + T_t, \qquad k > 0 \tag{5}$$

2.1.2. Pose Calculation

As previously mentioned, the 3D mesh model of the object is predefined. Consequently, we can obtain the bounding box of the dynamic object and determine its maximum height H. We then compute the 3D coordinates of the intersection point $\boldsymbol{p}$ between the object's principal ray $\bar{r}^{M_t}$ and the horizontal plane at $y = H/2$. The corresponding ray parameter k is determined by solving the ray–plane intersection:

$$k^* = \frac{(H/2) - o_{w,y}}{d_{w,y}} \tag{6}$$

The full 3D position is then calculated as Eq 7:

$$\boldsymbol{p} = \bar{r}^{M_t}(k^*) \tag{7}$$

Finally, by taking only the x and z values from this point $\boldsymbol{p}$, we project it onto the ground plane ($y = 0$) to place the object.

The position calculation allows us to determine the object's placement for each frame. The object's direction of rotation is determined from the displacement vector $\boldsymbol{v}$, calculated as the difference between the current frame's position, $\boldsymbol{p}_t$, and the previous frame's position, $\boldsymbol{p}_{t-1}$.

$$\boldsymbol{v}_t = \boldsymbol{p}_t - \boldsymbol{p}_{t-1} = \langle x_t - x_{t-1}, 0, z_t - z_{t-1} \rangle \tag{8}$$

Although the object's position and rotation can be computed, significant inconsistencies may arise between consecutive frames if the masks are deformed or noisy. Such abrupt variations reduce fidelity, as the rotation calculation directly reflects them. To address this, we apply a stabilization filter composed of three components:

- **Motion gating / deadband:** Suppresses micro-jitters by treating negligible rotational changes as zero when motion is minimal.
- **Rate limiting:** Constrains the maximum rotation angle per frame, ensuring smooth and consistent turns.
- **Exponential moving average (EMA) smoothing:** Reduces noise by blending the newly computed orientation with the previously filtered orientation using spherical interpolation.

By applying this process to each frame, we obtain the object's position (x, z) and yaw rotation θ for the sequence. However, when the mask is completely missing, position and rotation cannot be computed for those frames. To maintain temporal consistency during such dropouts, we linearly interpolate both position and rotation across short gaps of up to N consecutive frames. Let $t_0 < t_1$ be the valid keyframes that bracket a gap of length $m = t_1 - t_0 - 1 \leq N$. For any missing frame $t \in (t_0, t_1)$, set $\lambda = \frac{t - t_0}{t_1 - t_0}$ and compute using Eqs. 9 and 10.

$$p_t = (1 - \lambda) p_{t_0} + \lambda p_{t_1} \tag{9}$$

$$\theta_t = \theta_{t_0} + warp_\pi(\theta_{t_1} - \theta_{t_0})\lambda \tag{10}$$

In Eq 10, $wrap_\pi(\cdot)$ maps angles to $(-\pi, \pi]$ to ensure shortest-arc interpolation. Interpolation of sections whose length exceeds N may cause problems such as objects penetrating the scene, so they are not interpolated and are left as post-processing targets.

The full procedure is summarized in Algorithm 1.

Algorithm 1 Algorithm **F** for dynamic object pose estimation.

Require: image sequence $\mathcal{I}$, 3D scene mesh $\mathcal{S}$, object mesh O, Camera $C = (T, K)$
Ensure: object pose sequence $\mathcal{P}$

for $t \leftarrow 1$ **to** n **do**
  $M_t \leftarrow SAM2(I_t)$ //Mask Image From Segment Anything Model 2
  $\bar{r}^{M_t}(k) \leftarrow MaskProjection(M_t)$
  $H \leftarrow Height(O)$
  $k^* \leftarrow \frac{(H/2)-o_y}{d_y}$
  $p_t \leftarrow \bar{r}^{M_t}(k^*)$
  if $t > 1$ **then**
    $v_t \leftarrow p_t - p_{t-1}$
    $\theta_t \leftarrow Filtering(v_t, v_{t-1})$ //Filtering with EMA, rate limit, deadband, motion gate
  end if
  $P_t = \{x_t, 0, z_t, \theta_t\}$
end for
if $LostCount(\{P_t\}_{t_0<t<t_1}) \leq N$ **then**
  Interpolation(P_t)
end if
return $\mathcal{P}$

3. Evaluation

To evaluate the proposed methodology, we use a synthetic scene created in Unity. The object's position and rotation information is logged for each frame, and these data serve as the GT. The methodology is then assessed by comparing and analyzing two sets of data against the GT: the data obtained with the stabilization filter applied and the data obtained without it.

In Figure 2, during frames 0–20, the object moves short distances and performs specific actions while largely stationary. From frames 21–35, it moves backward. Subsequently, the object moves straight, turns to the right, and then to the left, before ending the sequence. The same positions are obtained with both the filtered and unfiltered methods; however, the unfiltered method exhibits highly sporadic rotational directions, whereas the filtered method maintains consistency. More detailed results are provided in Figure 3. As illustrated in Figure 4, a masking error occurs between frames 30 and 40, leading to a substantial position error in this interval. In addition, after frame 90, an occlusion is observed, resulting in tracking failure and a further increase in position error.

As shown in Figure 3, between frames 0 and 40—where the inter-frame trajectory distance is short and both in-place rotations and masking errors occur—the unfiltered method exhibits large rotational fluctuations, whereas the filtered method maintains narrower fluctuations, demonstrating robustness to noise. However, compared to the unfiltered method, the filtered method cannot immediately capture rapid directional changes due to the maximum rotation speed limit observed during the right/left turning section (frames 70–90).

Table 1 shows that applying the filter significantly reduces errors compared to the unfiltered method. In particular, for the maximum angular error (MaxAE), the unfiltered method produced a large error of approximately 179°, whereas the filtered method reduced this error to about 63°.

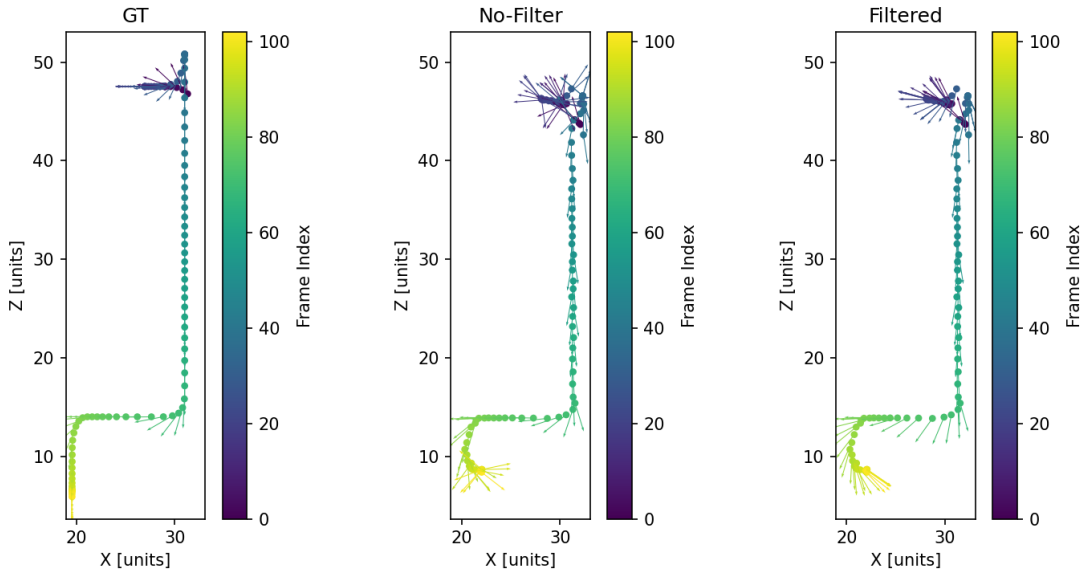

Figure 2: Scatter plot of object trajectories under three conditions: ground truth (GT), without stabilization filter, and with stabilization filter. Each dot represents the object's position, and arrows indicate the rotation direction at each frame.

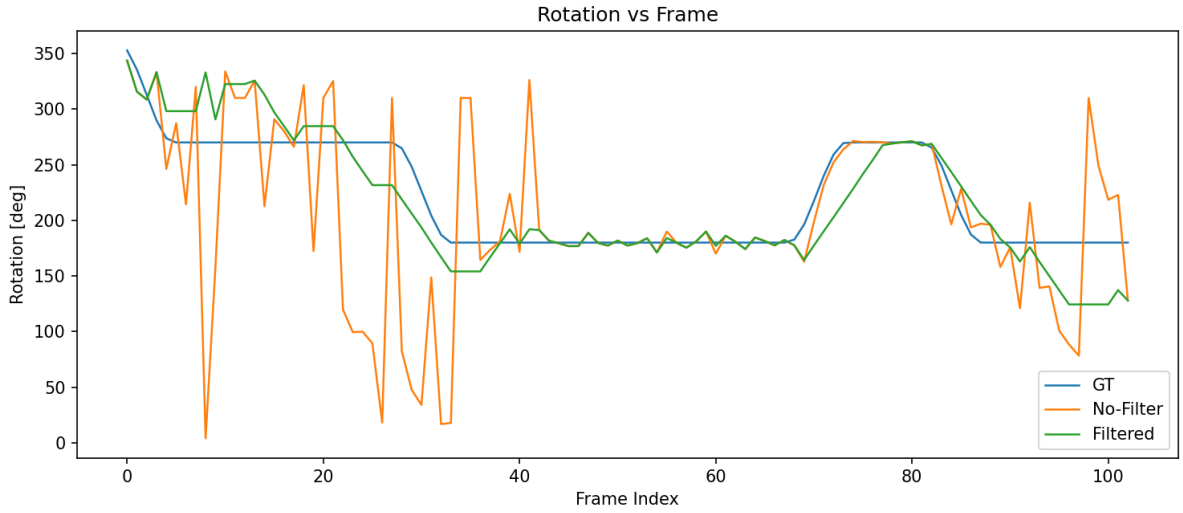

Figure 3: Yaw rotation values across frames for the ground truth (GT), filtered method, and unfiltered method.

4. Conclusion

This study proposes a lightweight pipeline that, after extracting masks using Segment Anything Model 2 (SAM2), performs mask projection, computes positions via the intersection between a principal ray and a plane, and approximates rotation (yaw) using frame-to-frame motion vectors. In addition, to suppress noises in the estimated rotation and ensure continuity along the time axis, we introduce a stabilization scheme that combines Gating, Deadband, Rate Limiting, and an exponential moving average (EMA). By incorporating this stabilization module, the system is designed to maintain spatiotemporal consistency even in the presence of noise and occasional errors. This design is practically meaningful in that it achieves computational efficiency suitable for real-time processing without complex optimization or large-scale learning.

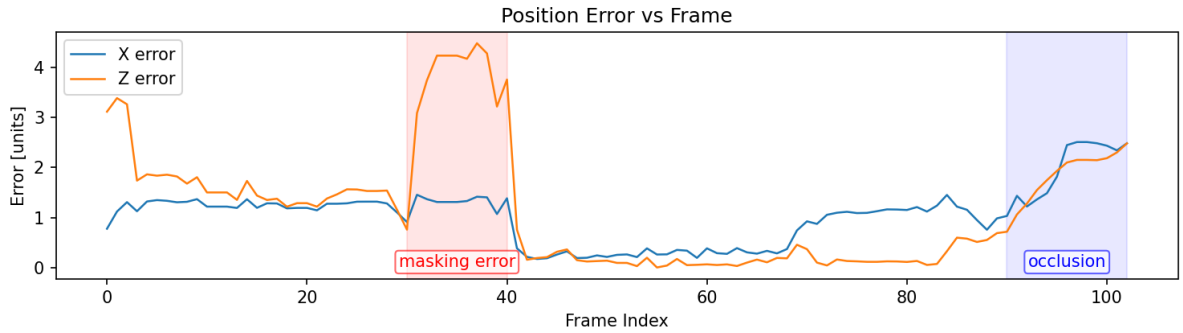

Figure 4: Frame-wise position error between the ground truth (GT) and each method. Masking errors occur between frames 30–40, and occlusion after frame 90 further increases the error. The error is computed as the absolute difference |GT − method|.

Table 1
Rotation angle error comparison between filtered and unfiltered methods.

Method	RMSE	MAE	MaxAE
No-Filter	66.41	41.52	179.46
Filtered	27.83	20.69	62.93

Nevertheless, the proposed approach is structurally dependent on segmentation quality. Because position and rotations are determined from masks produced by SAM2, a basic level of error is inherent, and large errors may occur when occlusions are present or when SAM2 fails due to its performance limits. Moreover, since rotation is determined by the motion vectors, it is difficult to correctly reflect orientation in scenarios dominated by lateral or backward motion, in-place rotation or in-place actions. Our method also assumes that objects remain in contact with the ground and therefore estimates only 3DoF (planar position and yaw); accordingly, it is not applicable to aerial objects (e.g., drones) or to objects exhibiting substantial pitch/roll variations. To address these structural issues, future work should introduce more robust methods for position and rotation estimation and extent the framework to full 6DoF pose estimation.

Furthermore, it operates under the assumption that the camera extrinsics in the digital twin coordinate system are estimated with very high accuracy through visual localization. However, even a small pose error can bias the principal ray-plane intersection, inducing position and rotation drift. To mitigate this, pose-stabilization strategies—such as drift compensation using semantic landmarks and sensor fusion with additional modalities (e.g., IMU)—should be considered. For dynamic object models with large intra-class shape variation, the fixed height assumption may not be valid if the shape dispersion is large, a fixed-height assumption may be invalid, potentially distorting position and orientation estimates. Future work should estimate object height online from frame-by-frame observations to preserve robustness when object models are inaccurate.

The proposed method was evaluated only in a synthetic virtual scene using quantitative metrics. For future work, in-the-wild validation is needed by applying the method to real video within a digital twin constructed from a 3D reconstruction of the physical environment. It is desirable to conduct multi-site, multi-scenario experiments spanning diverse indoor locations, camera setups, and object categories, and to complement them with user studies that qualitatively assess the temporal consistency of dynamic-object trajectories. The qualitative evaluation can use panel-based Likert-scale ratings or pairwise comparisons. Raters inspect side-by-side overlays on the source video and top-down trajectory visualizations, and statistical significance is assessed using appropriate tests. Such a combined quantitative–qualitative evaluation in real settings would allow a more rigorous demonstration of the generalizability and robustness of the proposed method.

In summary, the proposed method presents a concise and portable foundation that goes beyond the visualization of static structures in Digital Twins and aims for high-fidelity dynamic reproduction approaching functional equivalence for dynamic objects in scenes. Its significance lies in providing a balanced trade-off among lightweight implementation, real-time performance, and consistency in application domains dominated by dynamic factors-such as manufacturing, logistics, and smart cities. By pursuing the aforementioned extensions, we expect to progressively resolve challenges such as occlusion and in-place motion, thereby further improving the reliability and applicability of dynamic Digital Twin implementations.

Acknowledgments

This work was supported in part by the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2024-00344883)

Declaration on Generative AI

During the preparation of this work, the author(s) used GPT-5 in order to: Grammar and spelling check. The author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] ISO/IEC, Digital twin — concepts and terminology, International Standard ISO/IEC 30173:2023, 2023. URL: https://www.iso.org/standard/81442.html, standard.

[2] National Academies of Sciences, Engineering, and Medicine, The digital twin landscape, in: Foundational Research Gaps and Future Directions for Digital Twins, National Academies Press (US), 2024. URL: https://www.ncbi.nlm.nih.gov/books/NBK605499/.

[3] A. Fuller, et al., Digital twin: Enabling technologies, challenges and open research, IEEE Access 8 (2020) 108952–108971. doi:10.1109/ACCESS.2020.2998358.

[4] D. Jones, C. Snider, A. Nassehi, J. Yon, B. Hicks, Characterising the digital twin: A systematic literature review, CIRP Journal of Manufacturing Science and Technology 29 (2020) 36–52. URL: https://www.sciencedirect.com/science/article/pii/S1755581720300110. doi:https://doi.org/10.1016/j.cirpj.2020.02.002.

[5] D. M. Botín-Sanabria, A.-S. Mihaita, R. E. Peimbert-García, M. A. Ramírez-Moreno, R. A. Ramírez-Mendoza, J. d. J. Lozoya-Santos, Digital twin technology challenges and applications: A comprehensive review, Remote Sensing 14 (2022). URL: https://www.mdpi.com/2072-4292/14/6/1335. doi:10.3390/rs14061335.

[6] P. Muñoz, Measuring the fidelity of digital twin systems, in: Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings (MODELS '22), Association for Computing Machinery, New York, NY, USA, 2022, pp. 182–188. doi:10.1145/3550356.3558516.

[7] Digital Twin Consortium, Digital twin consortium defines digital twin, 2020. URL: https://www.digitaltwinconsortium.org/2020/12/digital-twin-consortium-defines-digital-twin/, accessed: 2025-08-21.

[8] F. Tao, M. Zhang, Y. Liu, A. Y. C. Nee, Digital twin driven prognostics and health management for complex equipment, CIRP Annals 67 (2018) 169–172. doi:10.1016/j.cirp.2018.04.055.

[9] M. S. Irfan, S. Dasgupta, M. Rahman, Toward transportation digital twin systems for traffic safety and mobility: A review, IEEE Internet of Things Journal 11 (2024) 24581–24603. doi:10.1109/JIOT.2024.3395186.

[10] Q. Picard, S. Chevobbe, M. Darouich, J.-Y. Didier, A survey on real-time 3d scene reconstruction with slam methods in embedded systems, arXiv preprint arXiv:2309.05349 (2023). arXiv:2309.05349.

[11] Y. Dai, Z. Hu, S. Zhang, L. Liu, A survey of detection-based video multi-object tracking, Displays 75 (2022) 102317. doi:10.1016/j.displa.2022.102317.

[12] C. Kober, M. Fette, J. P. Wulfsberg, A method for calculating optimum digital twin fidelity, Procedia CIRP 120 (2023) 1155–1160. URL: https://www.sciencedirect.com/science/article/pii/S2212827123008739. doi:https://doi.org/10.1016/j.procir.2023.09.141, 56th CIRP International Conference on Manufacturing Systems 2023.

[13] J. L. Schönberger, J.-M. Frahm, Structure-from-motion revisited, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 4104–4113. doi:10.1109/CVPR.2016.445.

[14] Y. Furukawa, J. Ponce, Accurate, dense, and robust multiview stereopsis, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2010) 1362–1376. doi:10.1109/TPAMI.2009.161.

[15] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, R. Ng, NeRF: Representing scenes as neural radiance fields for view synthesis, Communications of the ACM 65 (2022) 99–106. doi:10.1145/3503250.

[16] B. Kerbl, G. Kopanas, T. Leimkühler, G. Drettakis, 3d gaussian splatting for real-time radiance field rendering, ACM Transactions on Graphics 42 (2023) 1–14. doi:10.1145/3592433.

[17] C. Chen, B. Wang, C. X. Lu, N. Trigoni, A. Markham, Deep learning for visual localization and mapping: A survey, IEEE Transactions on Neural Networks and Learning Systems 35 (2024) 17000–17020. doi:10.1109/TNNLS.2023.3309809.

[18] W. Xiao, R. Chierchia, R. S. Cruz, X. Li, D. Ahmedt-Aristizabal, O. Salvado, C. Fookes, L. Lebrat, Neural radiance fields for the real world: A survey, 2025. URL: https://arxiv.org/abs/2501.13104. arXiv:2501.13104.

[19] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al., Segment anything, in: Proceedings of the IEEE/CVF international conference on computer vision, 2023, pp. 4015–4026.

[20] N. Ravi, et al., Sam 2: Segment anything in images and videos, arXiv preprint arXiv:2408.00714 (2024). arXiv:2408.00714.

[21] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, T.-K. Kim, Multiple object tracking: A literature review, Artificial Intelligence 293 (2021) 103448. URL: https://www.sciencedirect.com/science/article/pii/S0004370220301958. doi:https://doi.org/10.1016/j.artint.2020.103448.

[22] J. Shin, N. Hassan, A. S. M. Miah1, S. Nishimura, A comprehensive methodological survey of human activity recognition across divers data modalities, 2024. URL: https://arxiv.org/abs/2409.09678. arXiv:2409.09678.

[23] Z. Li, X. Wu, H. Du, F. Liu, H. Nghiem, G. Shi, A survey of state of the art large vision language models: Alignment, benchmark, evaluations and challenges, 2025. URL: https://arxiv.org/abs/2501.02189. arXiv:2501.02189.