International Metaverse Association

Metaverse Convergence Research Institute

ISSN 3059-0442

Journal of Virtual Convergence Research

Volume 2 Number 2 Apr. 2026

Received: 24 February 2026. Accepted: 29 April 2026

Journal of Virtual Convergence Research
2026, VOL. 2, NO. 2, 1~22.
https://intlmetaverse.com/home/journal?lang=en

International Metaverse Association &
Metaverse Convergence Research Institute

Dual Azure Kinect Laser Detection System for Tactical Training Simulation

Jaehyeon Park, Sanghun Park*

* Corresponding author: Sanghun Park. Email: mshpark@sogang.ac.kr

Abstract

This paper proposes a real-time system utilizing dual Azure Kinect DK cameras (Microsoft, 2023) to detect green laser pointers and transmit 3D vectors to a Unity simulation environment. To address the high cost and spatial constraints of existing infrared sensor-based tactical training systems, we present a low-cost alternative using computer vision algorithms. The system employs a dual-camera architecture where Camera 0 detects laser points based on Frame Difference and Green Excess weighted scores, while Camera 1 tracks hand and muzzle positions using Frame Difference and Depth Masking to eliminate interference. Detected 2D coordinates are converted into 3D vectors through screen homography or stereo calibration and transmitted to Unity via a UDP JSON protocol. In Unity, a thread-safe Producer-Consumer pattern is implemented to handle data reception, enabling Physics.Raycast-based target collision detection and LineRenderer visualization. The current production pipeline operates in a single-user mode using frame-difference detection, while the advanced modules (IR+HSV dual filtering, SORT-based tracking, and Mediapipe-based hand tracking) are validated at the component level on synthetic data and are reserved for an upcoming integrated MVP. This study therefore reports a system-level engineering study: it establishes the dual-camera architecture and the calibration / coordinate-transformation pipeline that any later integration will reuse, rather than claiming a fully validated end-to-end product.

Keywords : Tactical Training Simulation, Azure Kinect, Laser Detection, Computer Vision, Unity 3D

Dual Azure Kinect Laser Detection System for Tactical Training Simulation

1. Introduction

In recent years, the integration of physical tactical training with virtual simulation environments has become a critical requirement for modern military and police training. Systems like Virtra (2023) provide immersive experiences where trainees interact with projected scenarios using laser-equipped firearms. A core technical challenge in these simulators is the real-time, accurate detection of laser impact points on a large screen and reflecting them within a 3D engine such as Unity.

Traditionally, commercial solutions utilize infrared (IR) sensor panels to detect laser coordinates. While accurate, these hardware-based systems are prohibitively expensive, often exceeding $10,000, and suffer from rigid constraints regarding screen size and installation flexibility. Consequently, there is a growing demand for cost-effective alternatives that can maintain high fidelity without relying on specialized sensor hardware.

This paper proposes a novel laser detection and tracking system utilizing two Azure Kinect DK cameras and computer vision algorithms, reducing the hardware cost to approximately $800. By separating the detection tasks—one camera for laser points and another for hand/muzzle tracking—we address interference issues common in single-camera setups.

Comparison with prior tactical-laser systems

Camera-based laser pointer interaction has a long history. Olsen and Nielsen

(2001) and Soetedjo et al. (2014) proposed monocular camera-based laser interaction systems primarily for projector-screen UI; while elegant, those systems neither separate the trajectory origin (hand or muzzle) from the impact point on the screen nor reconstruct a 3D ballistic vector, limiting them to 2D screen-space hit registration. Thakur et al. (2022) addressed ambient-light tolerance on curved multi-projector displays but assumed a fixed-viewpoint user, again without recovering a 3D firing direction. Commercial tactical simulators such as VirTra (2023) achieve high accuracy through dedicated IR sensor panels at hardware costs exceeding $10,000 and with rigid screen-size constraints. In contrast, the system proposed here (i) functionally separates the trajectory endpoints across two depth cameras, (ii) reconstructs a 3D ballistic vector V = P1 − P2 in metric space, and (iii) does so on Commercial Off-The-Shelf hardware at roughly $800 — a combination that, to the best of our knowledge, has not been reported in the prior tactical-laser-simulation literature.

The main contributions of this study are as follows:

1. Dual Camera Architecture: We propose a separated detection pipeline where Camera 0 tracks the laser and Camera 1 tracks the user's hand, minimizing false positives.
2. Hybrid Detection Algorithm: We introduce a robust detection method combining Frame Difference with a Green Excess weighted score and an exponential decay spike map, enabling the detection of fleeting laser signals.
3. Dual Coordinate Transformation: The system supports both Homography-based mapping (2D to Screen) and Stereo Calibration (2D to 3D space), allowing for versatile deployment.

4. Thread-Safe Unity Integration: We present a reliable communication architecture using a Producer-Consumer pattern to parse UDP JSON data in a background thread, ensuring smooth visualization in the Unity main thread.

5. Automated Target Interaction: A physics-based raycasting system is implemented to automatically determine target hits and calculate scores within the simulation.

6. Taken together, this work positions itself primarily as an engineering systems contribution: it shows that a tactical laser-detection pipeline traditionally bound to dedicated IR hardware can be reconstructed on COTS depth-camera infrastructure. The underlying algorithmic ideas (hybrid IR+HSV detection fused with green-excess scoring, depth-gated hand tracking, and the dual-camera ballistic-vector formulation) are validated at the component level and offered as a foundation for subsequent end-to-end empirical study rather than as fully validated stand-alone scientific results.

2.**Theoretical Background**

2.1 **Vision-Based Hybrid Detection Algorithms**

Detecting a fleeting 532nm green laser pulse within a dynamic projection environment requires a robust fusion of spatio-temporal and spectral analyses. The proposed system models this detection process through theoretical stages: temporal persistence, chromatic filtering, and multi-factor confidence evaluation.

2.1.1 **Temporal Persistence and Background Modeling**

To isolate moving projectiles while adapting to gradual lighting changes in the

simulation environment, we employ an Exponential Moving Average (EMA) background model. The background B_t at pixel (x, y) and time t is continuously updated using a retention coefficient of $\alpha = 0.95$:

$$B_t(x,y) = \alpha \cdot B_{t-1}(x,y) + (1-\alpha) \cdot I_t(x,y) \quad (1)$$

To overcome the limitations of simple frame differencing ($\Delta I_t = |I_t - B_{t-1}|$) due to sensor refresh rates, we introduce a Temporal Spike Map with exponential decay. After applying a 3×3 Gaussian blur to the difference frame ($\tilde{\Delta I_t}$), the spike map M_t is computed as:

$$M_t(x,y) = max(\tilde{\Delta I_t}(x,y), \gamma \cdot M_{t-1}(x,y)) \quad (2)$$

where the decay coefficient $\gamma = 0.88$ ensures that a transient spike retains over 50% of its peak intensity for approximately 5 frames.

2.1.2 **Chromatic Filtering and Adaptive Score Fusion**

To distinguish the 532nm laser from broadband ambient light, we calculate the Green Excess metric (E_c):

$$E_c(x,y) = C_G(x,y) - \frac{1}{2}(C_R(x,y) + C_B(x,y)) \quad (3)$$

Negative values are clamped to zero and smoothed with a 3×3 Gaussian kernel to yield $\tilde{E_c}$. The final detection candidates are derived through a weighted linear fusion of motion and color:

$$S(x,y) = 0.6 \cdot M_t(x,y) + 0.4 \cdot \tilde{E_c}(x,y) \quad (4)$$

Only pixels that pass an adaptive percentile threshold (top 99.5%)—which dynamically adjusts based on the mean scene intensity—and simultaneously satisfy $\tilde{E_c} \geq 25$ are

registered as valid candidates.

2.1.3 **Inverse Homography Screen Masking and Multi-Factor Confidence Evaluation**

To fundamentally block out-of-screen noise, an inverse homography matrix (H^{-1}) is used to transform the four corner points of the physical screen into camera image pixel coordinates, generating a polygon mask.

Filtered candidates are then evaluated by summing four independent factors: peak intensity (0–40%), local contrast (0–30%), proximity to the center (0–20%), and movement stability (0–10%). Only candidates with a total confidence score $C_{laser} \geq 0.3$ are confirmed as final impact coordinates.

2.1.4 **Depth-Gated Hand Tracking**

The auxiliary camera (Camera 1) applies a depth gate to prevent mistaking screen laser reflections for hand movements:

$$D_{mask}(x, y) = 1 \text{ if } 400\text{mm} \leq Z(x, y) \leq 2000\text{mm}, \text{ else } 0 \quad (5)$$

This mask is AND-operated with the frame difference to isolate only the actual user's movements located at close range. An outlier-robust hand depth z is extracted by taking the median of a 5×5 patch in the detected region.

Rationale for the depth bounds

The lower bound is set at 400 mm to align with the reliable minimum measurement range of the Azure Kinect ToF sensor in its NFOV operation mode (Microsoft, 2023); below this distance, depth readings exhibit significant noise and drop-out and are unsuitable for stable hand tracking. The upper bound of 2000 mm is determined by the

system's geometric configuration: in our setup the auxiliary camera (Camera 1) sits roughly 1.5 m from the standing user, while the projection screen lies further downstream and the two cameras are separated by a baseline of approximately 6 m (cf. Sections 3.1 and 4.1). Restricting Z(x,y) to within 2 m therefore excludes pixels whose depth corresponds to the screen surface or to retro-reflected laser energy, suppressing the most common source of false-positive hand activations in this layout. Together, the two bounds isolate a near-field volume in which only the user's body and the firearm/laser-equipped hand are expected to appear, decoupling hand motion from screen-plane interference. These bounds were chosen during the calibration described in Section 4.1 and are reported here as a heuristic that is specific to the present geometry rather than a universal sensor parameter.

2.2 **Dual-Camera Hybrid 3D Tracking Geometry**

Rather than relying on traditional triangulation based on epipolar geometry, this system integrates two independent objects (laser and hand) into a single 3D space by combining a depth sensor, homography transformation, and rigid body transformation based on extrinsic parameters, as illustrated in Figure 1.

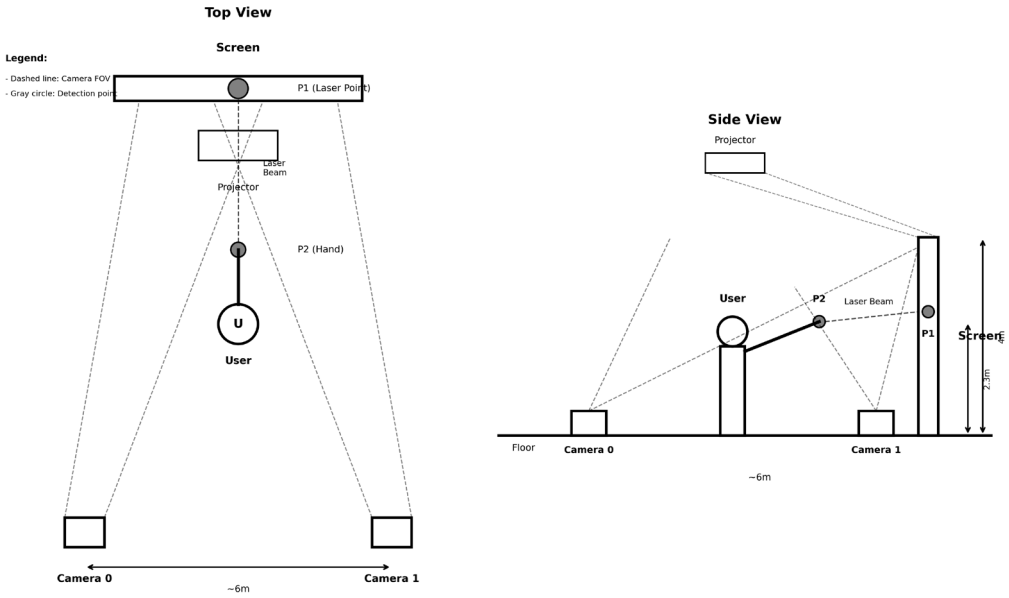


Figure 1. *Geometric model of the physical setup. The top and side views demonstrate the 3D ray-plane intersection geometry for Camera 0 and the stereo coordinate transformation for Camera 1.*

2.2.1 **Lens Distortion Correction and Homography-Based Ray-Plane Intersection (Camera 0)**

Camera 0 (tracking the laser) corrects lens distortion using the Brown-Conrady model to generate a 3D ray vector $\vec{r}$ in the normalized camera coordinate system. The intersection point X_{target} where this ray from the camera origin C intersects the screen plane Π is calculated as:

$$t = \frac{\vec{n}\cdot(P_0 - C)}{\vec{n}\cdot\vec{r}}, X_{target} = C + t\cdot\vec{r} \quad (6)$$

The mathematical model of the screen plane is defined in physical meters via a 3×3 homography matrix H calculated using the Direct Linear Transform (DLT).

2.2.2 **Extrinsic Parameter-Based Stereo Coordinate Transformation (Camera 1)**

Camera 1 (tracking the hand/muzzle) reconstructs the camera-local 3D point X_2 based on the distance z extracted from the depth sensor. For frames where depth data is lost, the system applies a hardcoded fallback distance (1500mm) to maintain stability. The reconstructed point X_2 is transformed into the reference coordinate system of Camera 0 via a rotation matrix R and translation vector T obtained from prior checkerboard calibration:

$$X_1 = R^T \cdot (X_2 - T) \tag{7}$$

2.2.3 **3D Ballistic Vector Computation and IPC Synchronization**

Independently detected X_{target} and X_{muzzle} data are matched via inter-process communication (JSON IPC) when their timestamp difference is within 1.0 second. The final ballistic trajectory vector $\vec{v}$ is computed as the Euclidean difference between the two positions integrated into a single coordinate system:

$$\vec{v} = X_{target} - X_{muzzle}, \hat{d} = \frac{\vec{v}}{||\vec{v}||+\epsilon} \tag{8}$$

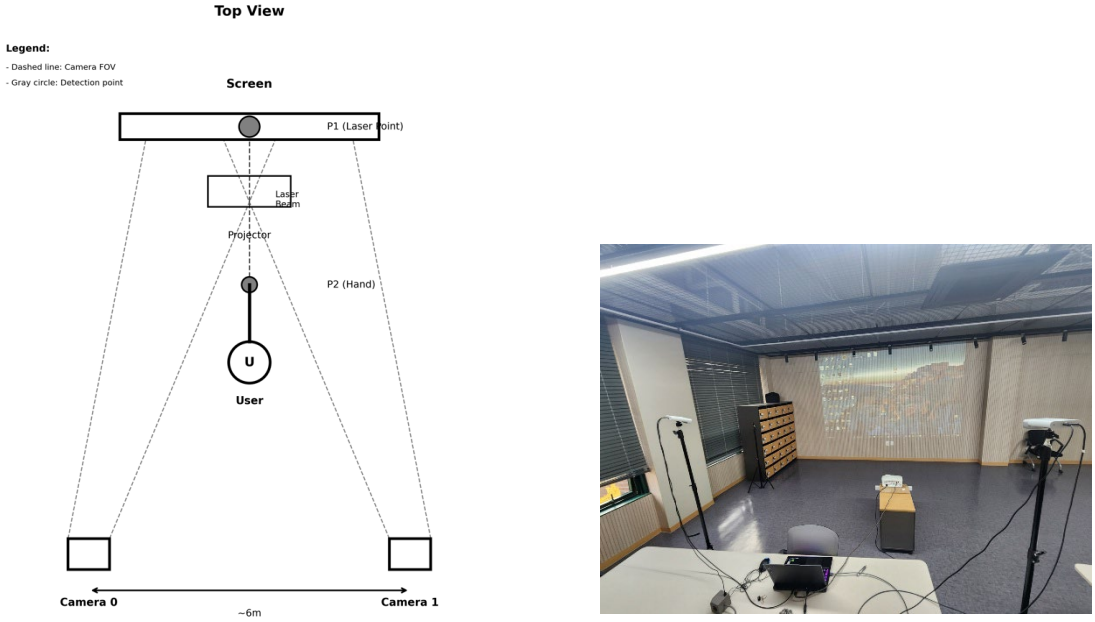
This hybrid pipeline provides a cost-effective alternative that overcomes the structural limitations of commercial cameras, supplying stable 3D firing vectors to the training simulator (Unity) without expensive dedicated sensor systems.

3.**Proposed System Architecture and Methodology**

3.1 **System Overview and Hardware Configuration**

The proposed system is designed to track a green laser pointer and the user's hand/muzzle position in real-time, mapping these inputs into a 3D simulation

environment. The hardware configuration consists of two Azure Kinect DK cameras, a 532nm green laser pointer, and a $3.73\text{m} \times 2.26\text{m}$ projection screen. Camera 0 is positioned to face the screen to detect laser impacts, while Camera 1 is oriented towards the user to track movements (see Figure 2).



(a) *(b)*

Figure 2. *Hardware configuration of the proposed dual-camera system. (a) Top-view schematic illustrating camera placements (~6m baseline) and field of view. (b) Actual laboratory implementation.*

3.2 **Dual-Camera Calibration Protocol**

To ensure accurate spatial mapping, the system employs a two-tier calibration protocol. First, a planar homography matrix (H) for Camera 0 is established by manually registering the four corners of the projection screen, which allows for the direct transformation of 2D image pixels into physical screen coordinates (in meters). Second, the extrinsic parameters (Rotation matrix R and Translation vector T) between Camera 0

and Camera 1 are acquired through a rigid body stereo calibration process using a checkerboard pattern, conducted over 22 iterative sessions to ensure stability. The comprehensive software pipeline, from raw detection to Unity raycast visualization, is summarized in Figure 3.

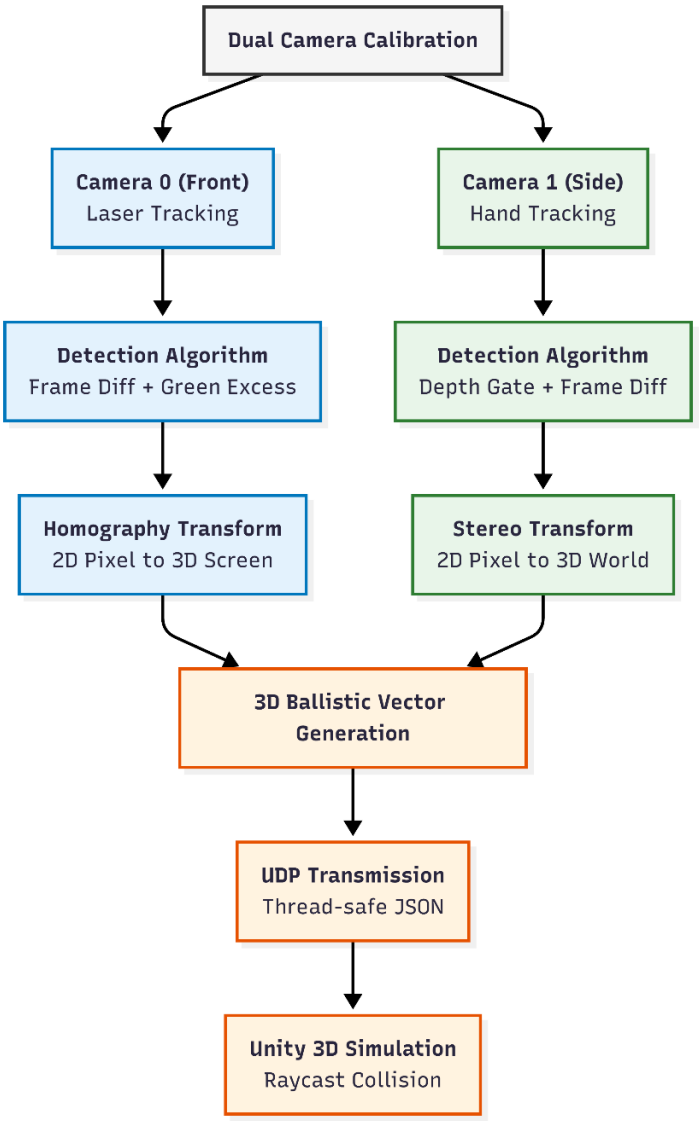


Figure 3. *Flowchart of the 3D ballistic vector generation process, detailing the hybrid coordinate transformation and the synchronization pipeline toward the Unity simulation engine.*

3.3 **Real-Time Inter-Process Communication (IPC)**

The detection pipeline operates across separate Python processes to maximize computational efficiency. Data synchronization between the laser tracking process and the hand tracking process is achieved via a JSON-based Inter-Process Communication (IPC) mechanism with atomic file writing to prevent data corruption. Once the temporal matching confirms a timestamp difference of less than 1.0 second, the computed 3D ballistic vector is transmitted to the Unity engine via a UDP protocol on port 9997.

3.4 **Thread-Safe Unity Integration and Visualization**

Integrating high-frequency UDP data into Unity poses a significant challenge, as Unity's API strictly prohibits execution outside the main thread. To resolve this, we implement a Thread-Safe message passing architecture. A background thread exclusively handles UDP reception and JSON parsing, utilizing a mutex lock (lock(msgLock)) to safely overwrite a shared variable with the latest data to prioritize real-time responsiveness over queued buffering. The main thread then consumes this data to perform safe API calls.

To align Python's right-handed coordinate system with Unity's left-handed system, a configurable flip flag matrix $F = \text{diag}(+1, -1, +1)$ is applied, specifically correcting the 180-degree Y-axis rotation inherent to the screen plane's orientation. Furthermore, an Exponential Moving Average ($\alpha = 0.2$) is applied to the camera vector to suppress orientation jitter.

Finally, collision detection is executed in a two-step software-hardware hybrid approach. A mathematical ray-plane intersection ($t = \frac{n \cdot (P_0 - S)}{n \cdot d}$) is first calculated to find the exact point on the projection screen. From 1cm beyond this intersection point, Unity's built-in Physics.Raycast is deployed to detect physical target collisions within the virtual environment up to 100 meters, subsequently triggering emission-based visual effects and

score calculations based on detection confidence and distance.

4.**Results and Component Validation**

4.1 **System Calibration and Precision**

The spatial accuracy of the proposed dual-camera system relies heavily on the calibration protocols. For Camera 0, a 4-point planar homography calibration was conducted on a 3.73m × 2.26m projection screen. The Direct Linear Transform (DLT) algorithm yielded an algebraic fit residual of approximately 10^{-6} px. While this indicates a mathematically exact fit for the four corner constraints, distortion correction was intentionally bypassed to optimize processing speed.

For the stereoscopic integration of Camera 1, 22 rigorous calibration sessions were performed using 20 checkerboard images (8 × 6 inner corners, 80mm square size) per session. The physical setup maintained a baseline of 2266.5mm and a convergence angle of 31.3°. The longitudinal calibration data showed consistent improvement, culminating in a highly stable production Stereo RMS error of 0.452 px, with an optimal historical record of 0.232 px. The physical experimental environment is shown in Figure 4.

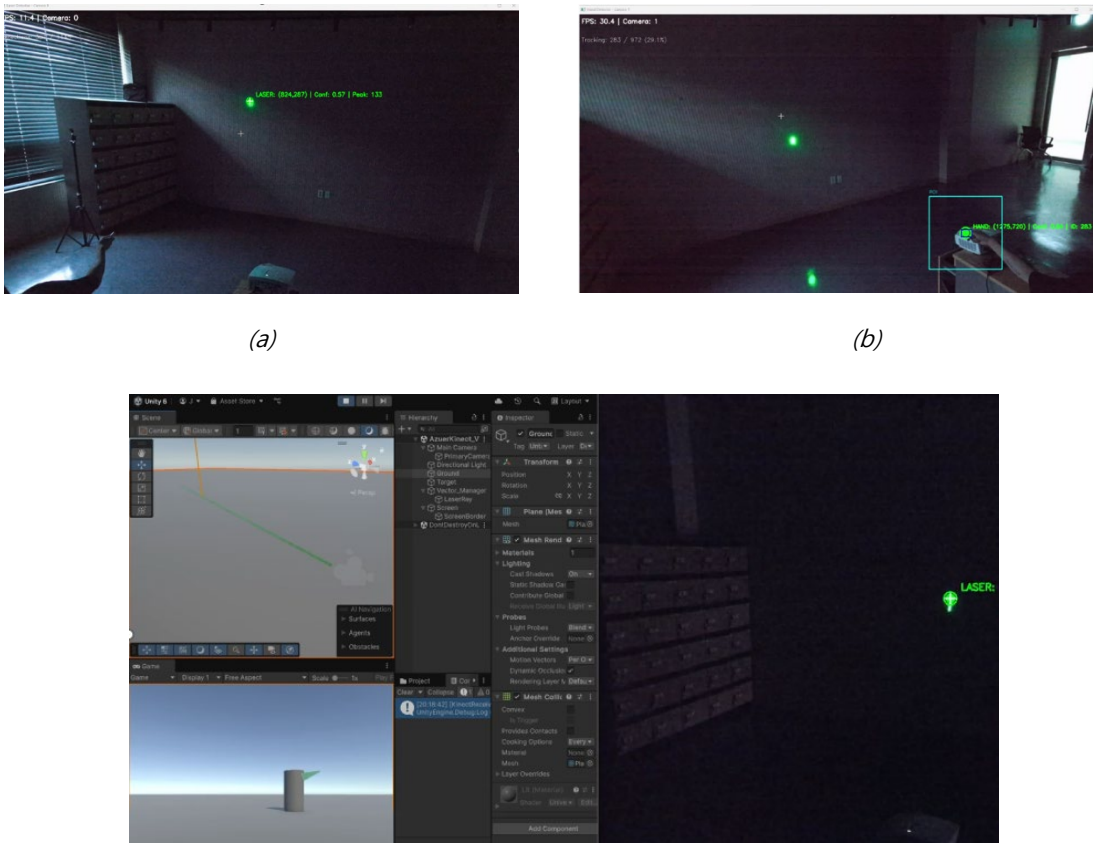
Figure 4. *Experimental environment. A participant aims a 532 nm green laser pointer at the projection screen while the Azure Kinect camera (on tripod, left) captures both IR and RGB data. The beam projector is visible in the foreground.*

4.2 **Algorithm Validation and Processing Performance**

As the integrated pipeline is currently in the MVP phase, component-level performance was validated using extensive synthetic datasets. The core IR+HSV detection module passed all 20 unit tests, successfully detecting stationary lasers—which traditional frame difference methods fail to capture—while robustly rejecting non-green light sources (red, blue, white, yellow). The module demonstrated stability across diverse ambient lighting conditions (intensity levels 30 to 200) and successfully identified up to 10 simultaneous laser points.

Performance metrics evaluated over 100 iterations of 1024 × 1024 synthetic IR and 1920 × 1080 RGB frames recorded an average processing time of 27.44 ms per frame for the

laser detection module. The Simple Online and Realtime Tracking (SORT) algorithm (Bewley et al., 2016), tested over 22 unit scenarios, maintained an ID consistency of over 70% across 50 frames and successfully recovered IDs after 3-frame occlusions. The SORT processing time remained under 5 ms for a single object and under 10 ms for 10 objects. To ensure real-time viability, the production system downscales input frames to a 960 px width. Representative outputs from each pipeline stage are presented in Figure 5.



(a) *(b)*

(c)

Figure 5. *Representative outputs from the integrated detection and visualization pipeline. (a) Camera 0: IR+HSV dual-filter detection output showing the detected laser position (304, 287) with a confidence score of 0.97 and peak IR intensity of 132. (b) Camera 1: Full-frame Mediapipe hand detection at 30.4 FPS with bounding-box annotation and two visible laser dots on the projection surface. (c) Unity 3D visualization showing*

Scene/Game views with ray rendering and receiver component configuration in the Inspector panel.

4.3 **Cost-Effectiveness Analysis**

Unlike conventional simulation systems that rely on dedicated hardware, the proposed architecture exclusively utilizes Commercial Off-The-Shelf (COTS) equipment.

Table 1. *Comparative Analysis of Hardware and Deployment Characteristics*

Feature	**Conventional Systems**	**Proposed Dual-Camera System**
Primary Sensor	Dedicated Infrared Sensor Arrays	COTS Depth Cameras (Azure Kinect)
Processing Method	Hardware-dependent	Software-based (Computer Vision)
Screen Constraint	Fixed to sensor array dimensions	Flexible (Homography adaptation)
System Upgrade	Requires hardware replacement	Algorithm update via software
Cost Efficiency	Prohibitively High	Highly Cost-Effective

As outlined in Table 1, the transition from hardware-bound sensing to software-based computer vision eliminates the need for expensive IR panels. This approach significantly lowers the entry barrier for tactical simulators while providing the flexibility to upgrade tracking capabilities (e.g., multi-user support) entirely through software iterations.

Production System Status

To avoid conflating component-level validation with deployed performance, we explicitly summarise the present integration status. The currently deployed (production) pipeline consists of: (i) the dual-camera hardware and calibration described in Sections 3.1 and 4.1, (ii) frame-difference-based laser detection on Camera 0 (single-user), (iii) depth-gated hand tracking on Camera 1, and (iv) the Unity-side thread-safe UDP receiver with Physics.Raycast collision. The advanced modules — IR+HSV dual filtering, the Kalman-

plus-Hungarian SORT tracker, and Mediapipe hand-landmark tracking — have been validated independently against synthetic data sets (Section 4.2) but are not yet wired into the live pipeline; their integration into a unified multi-user MVP is the immediate next milestone (Phase 1.4, see Section 5.3). All numerical results reported in this section should be read with this scope: calibration figures and per-frame processing times reflect the production pipeline, whereas tracking-related figures reflect component-level evaluations on synthetic inputs.

5.**Conclusion and Future Work**

5.1 **Summary of Contributions**

This study presented an engineering systems study of a cost-effective, dual-camera laser detection pipeline for tactical training simulations, in which the production pipeline and the component-level advanced modules are reported separately rather than as a single fully integrated product. The core contributions include a functional role separation architecture, mathematically rigorous 2D-to-3D coordinate mapping via planar homography, and stereo calibration achieving a production RMS error of 0.452 px. Furthermore, we implemented a thread-safe UDP communication protocol using a Producer-Consumer pattern for seamless Unity integration. To advance beyond basic detection, an IR+HSV dual-filter and a 7D Kalman-based SORT (Bar-Shalom et al., 2001; Kalman, 1960) tracker were designed and systematically validated through synthetic unit testing, demonstrating the algorithmic feasibility of accurate detection and multi-target tracking using consumer-grade depth cameras at a fraction of the cost of commercial infrared panels (approximately $800 per pair).

5.2 Current Limitations

Despite these architectural milestones, the current operational production system is limited to a single-user mode utilizing basic frame difference detection. Consequently, the system remains susceptible to false positives from dynamic backgrounds and cannot detect stationary lasers. Additionally, while the advanced modules (IR+HSV, SORT, and Mediapipe) have passed unit tests, they remain independent components; the full pipeline has yet to be integrated, meaning end-to-end latency and multi-user performance have not been measured in a physical environment. Finally, the 4-point homography calibration relies on an exact algebraic fit, lacking independent verification points to quantify sub-pixel accuracy across the entire screen.

5.3 Future Work and Roadmap

The immediate next step (Phase 1.4) is the integration of the independently validated IR+HSV, SORT, and Mediapipe hand-tracking (Lugaresi et al., 2019) modules into a unified MVP pipeline, enabling real-camera end-to-end performance benchmarking. To support multi-user environments, the system will employ the Hungarian algorithm (Kuhn, 1955) for optimal laser-hand assignment. For short-term improvements (Phase 2), a Trajectory Re-Identification (ReID) system is planned, which will utilize user-specific hand tremor frequencies (4–12Hz) and movement velocity features to increase ID consistency to a target of 90%.

In the long term (Phase 4), we propose "LaserNet," an end-to-end deep learning model featuring a dual-stream ResNet50 backbone, cross-attention fusion, and LSTM temporal modeling, aiming for over 95% ID consistency and real-time GPU inference. Ultimately, this progressive roadmap will establish a robust framework for law enforcement tactical

training, safely substituting live ammunition with precise green laser pointers in complex, multi-officer engagement scenarios.

Phase 2 (extended): quantitative end-to-end benchmarking and deployment readiness

In direct response to reviewer feedback, a controlled comparison study is planned as part of future research. Two reference baselines will be used: (a) a commercial IR-panel tactical simulator (VirTra-class) and (b) a monocular camera laser-pointer baseline, evaluated under matched lighting, screen size, and user pose. Reported metrics will include hit-detection accuracy expressed as point error against ground-truth panel readings, end-to-end latency from photon to Unity-side hit registration, and multi-user ID-consistency under cross-track scenarios. We also plan a deployment-readiness study covering ambient-lighting variability, the range of muzzle-to-screen geometries that the depth gate can support, and graceful degradation under partial occlusion. These studies directly address the practical-utility concerns raised during review and complement the component-level evaluation reported in Section 4.

Acknowledgment

This research was supported by the 2025 Metaverse Convergence Graduate School Program of the Ministry of Science and ICT and the Institute for Information & Communications Technology Planning & Evaluation (IITP) (RS-2022-00156318), and by the 2025 Culture Technology R&D Program of the Ministry of Culture, Sports and Tourism and the Korea Creative Content Agency (KOCCA) (RS-2023-00219237).

References

Bar-Shalom, Y., Li, X. R., & Kirubarajan, T. (2001). *Estimation with applications to tracking and navigation: Theory algorithms and software*. John Wiley & Sons.

Bernardin, K., & Stiefelhagen, R. (2008). Evaluating multiple object tracking performance: The CLEAR MOT metrics. *EURASIP Journal on Image and Video Processing*, *2008*, 1-10.

Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016). Simple online and realtime tracking. *2016 IEEE International Conference on Image Processing (ICIP)*, 3464-3468.

Collins, R. T., Lipton, A. J., & Kanade, T. (2000). Introduction to the special section on video surveillance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 745-746.

Hartley, R., & Zisserman, A. (2003). *Multiple view geometry in computer vision* (2nd ed.). Cambridge University Press.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, *82*(1), 35-45.

Kirstein, C., & Muller, H. (1998). Interaction with a projection screen using a camera-tracked laser pointer. *Proceedings IEEE International Conference on Multimedia Computing and Systems*, 191-192.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, *2*(1-2), 83-97.

Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., ... & Grundmann, M. (2019). MediaPipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*.

Marr, D. (2010). *Vision: A computational investigation into the human representation and processing of visual information*. MIT Press.

Merriaux, P., Dupuis, Y., Boutteau, R., Vasseur, P., & Savatier, X. (2017). A study of Vicon system positioning performance. *Sensors*, *17*(7), 1591.

Microsoft. (2023). Azure Kinect DK documentation. Retrieved from https://learn.microsoft.com/en-us/azure/kinect-dk/

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, *5*(1), 32-38.

Olsen, D. R., & Nielsen, T. (2001). Laser pointer interaction. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 17-22.

Radke, R. J., Andra, S., Al-Kofahi, O., & Roysam, B. (2005). Image change detection algorithms: A systematic survey. *IEEE Transactions on Image Processing*, *14*(3), 294-307.

Soetedjo, A., Mahmudi, A., Ashari, M. I., & Nakhoda, Y. I. (2014). Detecting laser spot in shooting simulator using an embedded camera. *International Journal on Smart Sensing and Intelligent Systems*, *7*(1), 423-441.

Stauffer, C., & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2*, 246-252.

Szeliski, R. (2022). *Computer vision: Algorithms and applications* (2nd ed.). Springer Nature.

Thakur, S., Urs, M., Ibrahim, M. T., Sidenko, A., & Majumder, A. (2022). Ambient light tolerant laser-pen based interaction with curved multi-projector displays. *Lecture Notes in Computer Science* (Vol. 13303). Springer.

VirTra. (2023). VirTra Tactical Training Simulators. Retrieved from https://www.virtra.com

Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. *2017 IEEE International Conference on Image Processing (ICIP)*, 3645-3649.