iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79333

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Automated 2D Blueprint-to-3D Gamified Tactical Training Simulator for Security Forces Using YOLOv8

Mahin Fareeha[1], Abdul Qadir Syed Quadri[2], Mohammed Shafi Omair[3], B.J. Job Karuna Sagar[4]
Department of Computer Science Engineering, Methodist College of Engineering and Technology, (Affiliated to Osmania University), Hyderabad, Telangana, India

Abstract: ***Security forces frequently utilize 2D architectural blueprint diagrams to plan operations within complex indoor environments. Nevertheless, understanding spatial information within a 2D blueprint under time constraints limits operational efficiency. This paper proposes an automated system that converts 2D blueprint diagrams into interactive 3D tactical training environments using a combination of deep learning and procedural generation techniques. The system employs a custom-trained model based on YOLOv8 to detect critical structural elements such as walls, doors, windows, staircases, and balconies within blueprint diagrams. The detected elements are then structured and integrated with a Unity-based engine to create an interactive 3D simulation environment in real time. The system also includes gamified mission scenarios, user interaction mechanisms, and performance analytics such as path efficiency, time efficiency, and task efficiency metrics. The entire system is implemented as an offline pipeline, ensuring efficient execution on conventional computing platforms. Experimental results demonstrate efficient detection performance and generation efficiency, thereby reducing time and cost compared to conventional physical model-based approaches. The proposed system presents a scalable solution for efficient tactical training environments.***

Index Terms: ***Blueprint analysis, deep learning, object detection, convolutional neural networks, YOLOv8, procedural generation, 3D environment reconstruction, tactical simulation, Unity 3D, computer vision, real-time systems, simulation-based training.***

I. INTRODUCTION

A. Motivation

Modern security forces, such as counter-terrorism units like the National Security Guard (NSG), Anti-Terrorism Squad (ATS), Rapid Action Force (RAF), SWAT, etc., often depend on two-dimensional (2D) architectural blueprints for mission planning and walkthroughs in complex urban environments. The use of 2D blueprints forces security forces to mentally recreate three-dimensional spatial relationships in a state of severe time constraint. This often leads to inadequate situational awareness, poor route planning, and increased risk of operations [1],[2].

Traditionally, security forces have employed physical mock-ups of buildings to be targeted. However, these mock-ups are expensive (₹50,000 to ₹2.5 lakh per environment) and time-consuming (3 to 7 days per environment), offering limited scalability. Moreover, these mock-ups cannot be easily re-configured in response to changes in intelligence on terrorists [3],[4]. Recent research has validated that traditional lecture-based and static instructional methods continue to dominate security forces' and military training programs, while simulation-based gaming technology remains severely underutilized due to resource constraints and lack of accessible methodological frameworks [5].

High-fidelity simulation has been found to have a significant impact on decision-making, spatial understanding, and engagement. Zulick et al. have shown that by combining clinical healthcare professionals with experts from law enforcement in a simulation-based crisis management training program, there were improvements in police cadets' de-escalation skills [2]. Alshehhi et al. have also shown that serious game techniques can improve strategic thinking skills for senior police officers [3]. In the military domain, Kubola et al. have shown that the SUS score for their 3D serious game simulator was 79.5. The feedback from the users indicated that the serious game simulator was realistic, cost-effective, and could replace expensive live exercises [4]. Bojor has validated digital wargaming in ARMA 3 for developing skills in a risk-free environment [6]. Gamification frameworks such as MDA and Octalysis have been validated by Ivanjko et al. for improving decision-making in military higher education [7]. Batzos et al. have validated gamification and serious games for improving security awareness in the domain of cybersecurity and first-responder training [8].

Rakhmat and Fauzian have designed a Holomatrix-based mixed reality tactical combat simulator for brigade-level training in the military domain [9]. Steven et al. carried out a systematic review of the use of virtual reality technology in military warfare training and its integration with robotics, haptics, and signal processing [10]. Sinogina et al. also explored the potential of simulation and gaming technologies in the Russian curriculum for the Basics of Safety and Protection of the Motherland and found that the lack of resources is a barrier to the use of these technologies despite the positive outcomes of their use for the formation of practical skills and patriotic motivations of the learners [5].

Despite the positive outcomes of the use of these technologies for the formation of practical skills and patriotic motivations of the learners, none of the existing systems have the capability to ingest a newly received scanned 2D blueprint (JPG/PNG/PDF) and produce a fully walkable gamified 3D tactical training environment in a matter of minutes on a regular laptop computer. The advances in computer vision, especially YOLOv8 object detection algorithms, and game engines such as Unity have made this integration possible. The current work bridges the existing gap by providing an end-to-end fully offline AI solution for Indian security forces' training needs.

B. Background

Before the advent of digital simulation and gamification technologies, the training of security forces is carried out by traditional methods. The operators were given 2D architectural blueprints in the form of prints or scanned images. They were required to mentally visualize and memorize the architecture of buildings with many floors, room connectivity, and the location of possible threats, all within a very short time frame [1], [5]. This process is called manual spatial interpretation. It is extremely taxing for the operators, and the chances of errors during actual operations were high.

To overcome the drawbacks of using 2D blueprints, mock-ups of the buildings were prepared. They were designed using wood, concrete blocks, panels, and even shipping containers. It took around 3 to 7 days to prepare the mock-ups, which cost between ₹50,000 and ₹2.5 lakh per building. They could be prepared for only the most important operations. It was impossible to change the scenarios frequently. Live field exercises, force-on-force training, and lectures were the main training tools. Sinogina et al. [1] stated that lectures, talks, and theoretical demonstrations still comprise the majority of the training programs related to security. Practical simulation, role plays, quests, and virtual environments are used sporadically. This is because there is a severe lack of equipment, methodological guidelines, and infrastructure.

Even structured programs, such as the Memphis Model Crisis Intervention Team, required 40 hours of specific training per officer and relied on external partnerships with mental health experts, which are not always available in resource-constrained environments [2]. These traditional approaches offered little in terms of repeatability and provided no objective measures of performance (such as path efficiency or time optimization). As a result, security forces continued to face challenges in achieving consistency in spatial and tactical understanding as they moved from 2D diagrams to real-world scenarios.

C. Contributions

The system combines object detection using deep learning, procedural generation, and real-time gamified simulation.

The contributions are as follows:

(i) An end-to-end offline artificial intelligence system capable of ingesting scanned building diagrams (JPG, PNG, PDF), automatically detecting building structures (walls, doors, windows, staircases, balconies) using a custom-trained YOLOv8 model, and (ii) a modular Unity-based procedural reconstruction system (DetectionLoader, StructureBuilder, EnvironmentBuilder, SimulationManager) capable of recreating a physically accurate, navigable 3D environment with intelligent behaviors, (iii) a full-fledged gamified simulation system with real-time interaction, threat zone highlighting, breach simulation, footstep sounds, and comprehensive after-mission analysis (path efficiency, time efficiency, completion grade), (iv) a lightweight Flask-based backend system with a military-grade web dashboard for blueprint upload, manual labeling, trainee management, mission assignment, and live performance synchronization, all of which are performed completely offline, and (v) empirical validation through object detection metrics, usability studies, and feasibility analysis for NSG/ATS/RAF/SWAT training scenarios.

II. LITERATURE SURVEY

A. Related Works

Recent research in simulation-based training, 2D-to-3D reconstruction, and deep learning has shown considerable progress; however, there are certain limitations when these techniques are used in real-world tactical training.

1) Simulation-Based Tactical Training Systems

Research has been conducted on simulation-based training systems and serious game technology in military and law enforcement environments. Zulick et al. [1] and Alshehhi et al. [2] have shown positive results in decision-making skills and engagement in simulation-based learning systems. Kubola et al. [3] and Bojor [4] have shown the potential of game engines and digital wargaming environments in simulation-based training systems. Sinogina et al. [5] have highlighted the importance of simulation and gaming technology in military and law enforcement learning environments

2) 2D-to-3D Reconstruction and Building Modeling

Notable developments have been made regarding the automated reconstruction of 3D models from 2D inputs. For example, Barreiro et al. [6], Schuegraf et al. [7], Cai et al. [8], Deshmukh et al. [9], among others, have used deep learning techniques as well as scan-to-BIM approaches to attain high accuracy in the reconstruction process. Feist et al. [10] and Vinodkumar et al. [11] provide comprehensive reviews on the developments in the aforementioned domain. Nonetheless, the aforementioned approaches are mostly based on clean inputs such as CAD drawings, images, as well as LiDAR point clouds, which do not resemble real-world degraded blueprints. In addition, the aforementioned approaches do not provide interactive simulations, AI-based behavior, as well as gamification.

3) Blueprint Analysis and Object Detection

Convolutional neural networks as well as object detection techniques such as YOLOv8 are widely used in the analysis of architectural as well as electrical blueprints. For example, Tanaka et al. [12] attained high accuracy in the detection of electrical components in blueprints through the use of the aforementioned techniques. In another study, Fathima et al. [13] used an unsupervised autoencoder approach to attain high accuracy in converting 2D blueprints to 3D models. Nonetheless, the aforementioned approaches are mostly used in the analysis of blueprints as well as 2D semantic extraction.

4) Gamification and Serious Games

Studies such as those by Ivanjko et al. [14], Batzos et al. [15], Rakhmat and Fauzian [16] confirm the effectiveness of gamification in the enhancement of the learning process as well as the level of user interaction. In addition, the use of serious games is also effective, as noted by Steven et al. [17]. While these systems offer advantages, they are normally limited to generic environments, lack contextualisation in real-world environments, and frequently require hardware support such as AR/VR devices. They do not support automated generation of environments based on real data.

B. Research Gaps

Although the works discussed in the previous section have shown robust foundations in simulation-based training [1]–[5], 2D-to-3D reconstruction [6]–[8], [10], and blueprint analysis [9], there are some major gaps in these techniques when they are considered for practical use in tactical training of Indian security forces such as NSG, ATS, RAF, and SWAT.

The first major gap in these techniques is that none of the simulation-based training tools [1]–[5] have the capability to automatically ingest newly received scanned 2D blueprints in image formats such as JPG, PNG, or PDF, which are normally provided to the security forces during time-sensitive operations. All of these tools can only handle scenarios that are manually authored by the instructor or developer.

The second major gap in these techniques is that none of the 2D-to-3D reconstruction techniques [6]–[8], [10] can handle poor-quality blueprints, such as those normally provided to Indian security forces. All of these techniques have been shown to work on high-quality data such as vector-based CAD drawings, images from aerial cameras, or LiDAR point clouds. However, in the case of blueprints normally provided to Indian security forces, these images are normally of poor quality, such as scanned images of hand-written notes, faded images, or images with overlapping symbols. Thirdly, the focus of existing blueprint analysis methods such as those based on [9] is limited to 2D semantic extraction and stops short of expanding these capabilities into procedural 3D modeling and simulation. The transition from detection to a fully navigable 3D training space is a time-consuming process. Lastly, an end-to-end solution does not yet exist that integrates (i) YOLOv8-based object detection directly from real scanned blueprints, (ii) procedural 3D modeling and simulation inside a game engine such as Unity, and (iii) a comprehensive gamified tactical simulation environment that includes interaction, breach simulation, threat zone visualization, and performance analytics such as path efficiency, time efficiency, breach statistics, and after-action reporting. The absence of such a unified solution forces security forces to resort to traditional methods or use individual tools that are not integrated into a unified solution.

Lastly, most existing solutions are not deployable in the field due to requirements such as high-end GPU support, internet connectivity, commercial software licenses, and/or AR/VR headset requirements [1], [4], [5]. Such requirements make these solutions difficult to deploy for NSG, ATS, RAF, SWAT forces, and other security forces due to the requirement for training in the field on laptops without internet access.

C. The Proposed System

To fill this void, this work proposes a novel end-to-end, fully offline-based artificial intelligence system, which takes a scanned 2D blueprint image (in any format: JPG, PNG, PDF), detects structural elements using a YOLOv8-based object detection model, procedurally generates a realistic 3D tactical environment using Unity 6 (Universal Render Pipeline), and offers a gamified mission rehearsal tool with live interaction, breach simulation, threat zone highlighting, and quantitative performance analysis (path efficiency, time efficiency, after-action report).

The entire system is designed to work completely offline, without any internet or high-end hardware requirement, allowing security forces to convert any newly received blueprint image into a fully functional 3D environment in a matter of minutes.

This novel system directly addresses the identified shortcomings in existing literature by unifying blueprint analysis, 3D reconstruction, and tactical simulation in a single, specially designed tool for Indian security forces.

1) Blueprint-Based Environment Understanding

The first step in this work involves blueprint image analysis, where images are processed to identify structural elements such as walls, doors, and windows. Unlike other images, blueprint images contain symbolic representations, different scales, and noise such as stamps and handwritten text, which makes blueprint image analysis difficult [9], [12].

Recent studies on floor plan analysis using deep learning techniques have focused on floor plan image analysis using object detection and segmentation techniques to identify architectural elements in a blueprint image. However, existing techniques are mostly designed for clean data, which is not generally available in real-world scenarios [8], [10].

To overcome this, object detection techniques are used to identify architectural elements in a blueprint image in a more efficient manner [5]. Given a bounding box defined by coordinates (xmin, ymin, xmax, ymax) over an image of width W and height H, the normalized representation is computed as:

$$[x_c = \frac{x_{\min} + x_{\max}}{2W}, \quad y_c = \frac{y_{\min} + y_{\max}}{2H}]$$
$$[w = \frac{x_{\max} - x_{\min}}{W}, \quad h = \frac{y_{\max} - y_{\min}}{H}]\#(1)$$

These normalized parameters allow for consistent mapping across different resolutions of the blueprint and act as input to 3D reconstruction.

To avoid duplicate object detection, Non-Maximum Suppression (NMS) is performed using Intersection over Union (IoU), where IoU is given by:

$$[IoU = \frac{Area(B_1 \cap B_2)}{Area(B_1 \cup B_2)}]\#(2)$$

Only objects with a confidence level above a certain predefined value and those that meet IoU conditions will be considered.

2) 2D-to-3D Mapping and Procedural Reconstruction

The extracted object features are then mapped to a 3D coordinate system. The system normalizes 2D coordinates to Unity's world coordinate system, where the horizontal plane is represented along axes X-Z.

The transformation is given by:

$$[X = x_c \cdot S, \quad Z = y_c \cdot S, \quad Y = 0]\#(3)$$

where S represents a global scaling factor controlling the size of the generated environment.

Existing research on 3D reconstruction from 2D plans has concentrated on static building information modeling (BIM) generation or computer-aided design modeling, where inputs to such systems need to be clean [10], [11]. This is because such systems cannot handle interactive environments, making them unsuitable for training. This problem is solved using procedural generation, where objects such as walls, doors, and windows are created using detected object features [6].

To allow navigation within the generated environment, a navigation mesh, also known as NavMesh, is generated to allow both user-controlled and artificial navigation within the generated environment. This integration of object detection with procedural modeling ensures a fully automated system, where 2D input is used to generate a 3D interactive simulation environment [7], [13].

Performance Metrics and Evaluation

Simulation-based training is largely being used in military and law enforcement scenarios because it is effective in enhancing decision-making processes and provides a safe platform for training [1], [2]. However, the majority of these simulations lack adaptability to real-world scenarios because they are scenario-based or pre-defined environments [3], [4].

Gamification environments offer improved user engagement and retention because they add interactive features to the simulation environment [14]. Despite the advantages, the current environments lack the ability to integrate with real-world blueprint data and do not offer performance analytics.

To assess the effectiveness of the simulation, performance metrics are computed based on user interaction within the environment.

$$[Efficiency_{path} = \frac{D_{optimal}}{D_{actual}}]\#(4)$$

Similarly, time efficiency evaluates the speed of task completion:

$$[Efficiency_{time} = \frac{T_{optimal}}{T_{actual}}]\#(5)$$

These metrics can be used to evaluate the performance of the user objectively. which can address the current gaps in the literature [8], [9], [10].

III. PROPOSED FRAMEWORK

The framework consists of three major stages:

- blueprint detection and filtering;
- 2D-to-3D reconstruction and environment generation;
- simulation, evaluation, 3and deployment.

The goal is to transform a raw blueprint image (I) into an interactive 3D environment (E) suitable for tactical training. Each stage operates in a modular pipeline, ensuring flexibility, scalability, and ease of integration between components.

This structured approach enables efficient data flow from detection to simulation while maintaining real-time performance.

A. System Architecture

The system architecture is divided into four loosely coupled layers: (1) machine learning detection backend, (2) Flask REST API middleware, (3) Unity 3D procedural simulation engine, and (4) web-based command dashboard. Fig. 1 depicts the data flow between these layers.

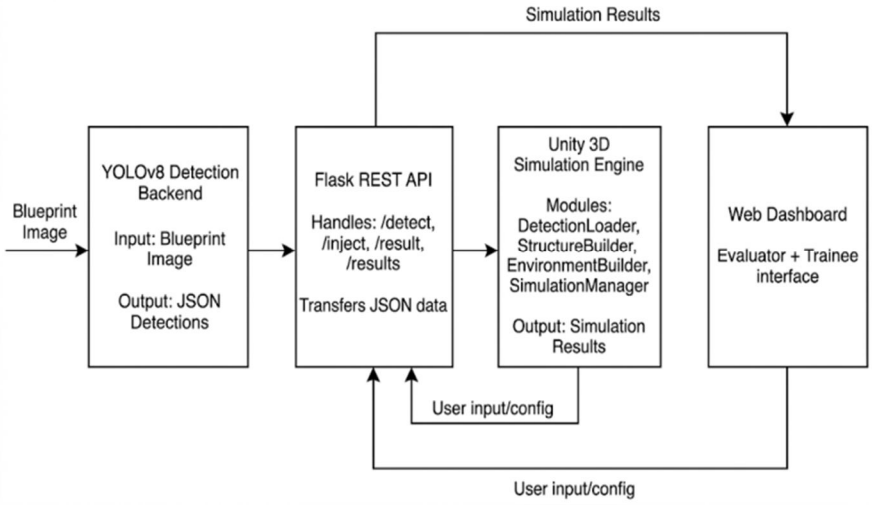


Fig. 1. system architecture: blueprint image flows through YOLOv8 detection, Flask API, Unity 3D procedural generator, and web dashboard.

1) YOLOv8 Detection Backend

The YOLOv8 model was fine-tuned on a custom dataset of 44 images of blueprint layouts with annotations using Roboflow for five classes: wall, door, window, staircase, and balcony.
Data augmentation techniques such as flip, rotation, brightness, and mosaic were used.
The model was trained on 100 epochs using an NVIDIA RTX 3050 GPU with a batch size of 16, image size 640x640.
The class-wise confidence thresholds were adjusted:
wall=0.25, door=0.25, window=0.40, staircase=0.01, balcony=0.30.
The lower confidence value for staircase class accounts for class imbalance.
Non-Maximum Suppression with IoU 0.25 and 0.04 distance on centers eliminates duplicate bounding boxes.
The final output consists of precise bounding box coordinates, class labels, and confidence scores for each detected element.
This structured detection data serves as the foundation for accurate 2D-to-3D conversion in the Unity simulation pipeline.

2) Flask REST API Middleware

A Flask 3.0 server has five endpoints:

- POST /detect: handles blueprint images and sends JSON data with normalized bounding box information, class label information, confidence information, and count information.
- POST /inject: combines ML results and manual annotations and sends data to Unity.
- POST /result: handles simulation results.
- GET /results: handles data requests from the dashboard.
- GET /health: handles server health checks.

The server automatically sends detections.json data to Unity's Resources folder.

3) Unity 3D Procedural Generation Pipeline

The Unity 6 (URP) pipeline is composed of four main modules:

- DetectionLoader: Loads JSON, applies thresholds, and performs NMS filtering. Removes small false positives (e.g., staircase noise).
- StructureBuilder: Generates 3D geometry (Walls extruded with thickness, doors with hinge-based animation, frames, and handles, windows with glass, sill, and frame, entrance door detected based on proximity to image edges
- EnvironmentBuilder: adds light sources, directional sun, ambient GI, fog, PBR materials, reflection probe, streetlights, exterior context.
- SimulationManager: manages pipeline and player spawning, integrates web dashboard.

4) Gamified Interaction Layer

- PlayerController enables first-person movement using Unity's Input System: WASD movement, mouse look, sprint, jump, and interaction keys. Players can open doors, interact with windows, and mark zones as safe or hostile. Audio feedback includes ambient sounds and footsteps.
- DoorController supports Standard, Locked, and Reinforced doors with multiple interaction attempts and smooth animation.
- WindowController allows interaction using E to inspect and triple F presses to break windows, triggering a glass-breaking effect where pieces fall apart, enhancing realism.
- GameManager tracks actions, maintains zone status, and computes metrics such as time efficiency, path efficiency, missed checkpoints, and a composite score, which is sent to Flask.

5) Web-Based Command Dashboard

The dashboard is a self-contained HTML-based dashboard that is HTML-based and has a military-themed dark color scheme and responsive design.
The Evaluator Portal supports blueprint upload, live detection visualization, manual annotation, confidence adjustment, trainee management, mission assignment, and environment configuration.
The capabilities of the Trainee Portal include Mission details, Performance (breakdown bars for best score / average score / path efficiency / time efficiency, grade distribution histogram, score trend chart), History (per-run table with grade, time, doors/windows/broken counts), and Rankings (animated podium for top 3, full leaderboard).

The results are updated from a Flask application every 10 seconds.

Algorithm 1 Blueprint-to-3D Tactical Simulation Pipeline

Require: Blueprint image I, trained model M, scaling factor S

D ← Detect(I, M)
D ← ApplyThresholds(D)
D ← NMS(D)
for each detection d_i ∈ D do
  Map coordinates to 3D
    $((X_i, Z_i) \leftarrow MapTo3D(x_i, y_i, S))$
  Scale dimension
    $((W_i, H_i) \leftarrow Scale(w_i, h_i, S))$
end for
Generate environment E
  $(E \leftarrow GenerateWalls(D))(E \leftarrow InsertDoorsWindows(E, D))(E \leftarrow AddStructures(E))$
Build NavMesh(E)
  $(\mathcal{N} \leftarrow BuildNavMesh(E))$
while simulation running do
  Capture user trajectory
  Compute efficiency metrics (η_p, η_t)
end while
Generate Performance Report

This combines the concepts of detection, geometric transformation, and procedural modeling in a unified framework, which facilitates fast creation of mission-oriented simulation environments directly from raw blueprints.

B. Methodology

The proposed methodology combines machine learning-based object detection, geometric transformation, procedural modeling, and real-time simulation in a unified framework.

The entire transformation process can be described as follows:

$$E = F(I)E = \mathcal{F}(I)E = F(I) \quad (6)$$

where I is the input blueprint image and E is the generated 3D simulation environment.

1) Blueprint Detection and Refinement

The first stage involves detecting structural components from the blueprint image using a trained YOLOv8 model. The detection output is represented as:

$$D = di = Detect(I, M) \quad (7)$$

where each detection $di = (c_i, b_i, s_i)$ contains class label, bounding box, and confidence score.

To improve reliability, detections are refined using class-specific thresholds and Non-Maximum Suppression (NMS):

$$D' = NMS(Threshold(D)) \quad (8)$$

The detection refinement process is summarized in Algorithm 2.

Algorithm 2 Detection Filtering and Refinement

Require: Raw detections D, confidence thresholds T_c, IoU threshold τ

$D_{filtered} \leftarrow \emptyset$
for each detection $d_i \in D$ do if confidence$(d_i) \geq T_c(class(d_i))$ then $D_{filtered} \leftarrow D_{filtered} \cup d_i$
end if
end for
$D_{final} \leftarrow \emptyset$
while $D_{filtered} \neq \emptyset$ do $d_{max} \leftarrow$ detection with highest confidence $D_{final} \leftarrow D_{final} \cup d_{max}$ remove all d_j where $IoU(d_{max}, d_j) \geq \tau$
end while
return D_{final}

2) Procedural 3D Environment Generation

The object refinement result is then converted to a 3D simulation environment using a procedural modeling approach. The bounding box is converted to a 3D object in the 3D environment using a mapping to world coordinates:

$$(x, y) \rightarrow (x \cdot S, y \cdot S) \quad (9)$$

where S is the scaling factor.
Walls are created using extrusion:

$$V = (x, y, z) \mid z \in [0, H] \quad (10)$$

where H represents wall height.
The navigation space is constructed, and a NavMesh is baked for realistic movement. The complete generation process is described in Algorithm 3.

Algorithm 3 Procedural 3D Environment Generation

Require: Filtered detections D, scale factor S, wall height H

Initialize empty environment E
for each detection $d_i \in D$ do
$(x, y, w, h) \leftarrow$ bounding box of d_i
$(x', y') \leftarrow (x * S, y * S)$
if class(d_i) == wall then
CreateWall$(x', y', w * S, H)$
else if class(d_i) == door then
CreateDoor$(x', y', w * S, H)$
else if class(d_i) == window then
CreateWindow$(x', y', w * S, H)$
else if class(d_i) == staircase then
CreateStaircase(x', y')
end if
end for
AddEnvironmentDetails(E)
BakeNavMesh(E)
return E

3) Simulation and Performance Evaluation

Once the environment is generated, real-time simulation is conducted using a first-person interaction model. Player actions drive state transitions:

$$s_{t+1} = T(s_t, a_t) \quad (11)$$

where s_t is the current state and a_t is the action taken.

Performance metrics are computed based on trajectory and mission objectives:

$$Score = w1 \cdot TimeEff + w2 \cdot PathEff - w3 \cdot Missed \quad (12)$$

where:

$TimeEff$: time efficiency

$PathEff$: path efficiency

$Missed$: missed objectives

The evaluation process is summarized in Algorithm 4.

Algorithm 4 Tactical Performance Evaluation

Require: Player trajectory T, mission objectives O

$time_{eff} \leftarrow$ ComputeTimeEfficiency(T)

$path_{eff} \leftarrow$ ComputePathEfficiency(T)

missed ← CountMissedObjectives(T, O)

$score \leftarrow w1 * time_{eff} + w2 * path_{eff} - w * missed$

if score ≥ $threshold_{high}$ then

grade ← "A"

else if score ≥ $threshold_{mid}$ then

grade ← "B"

else

grade ← "C"

end if

return score, grade

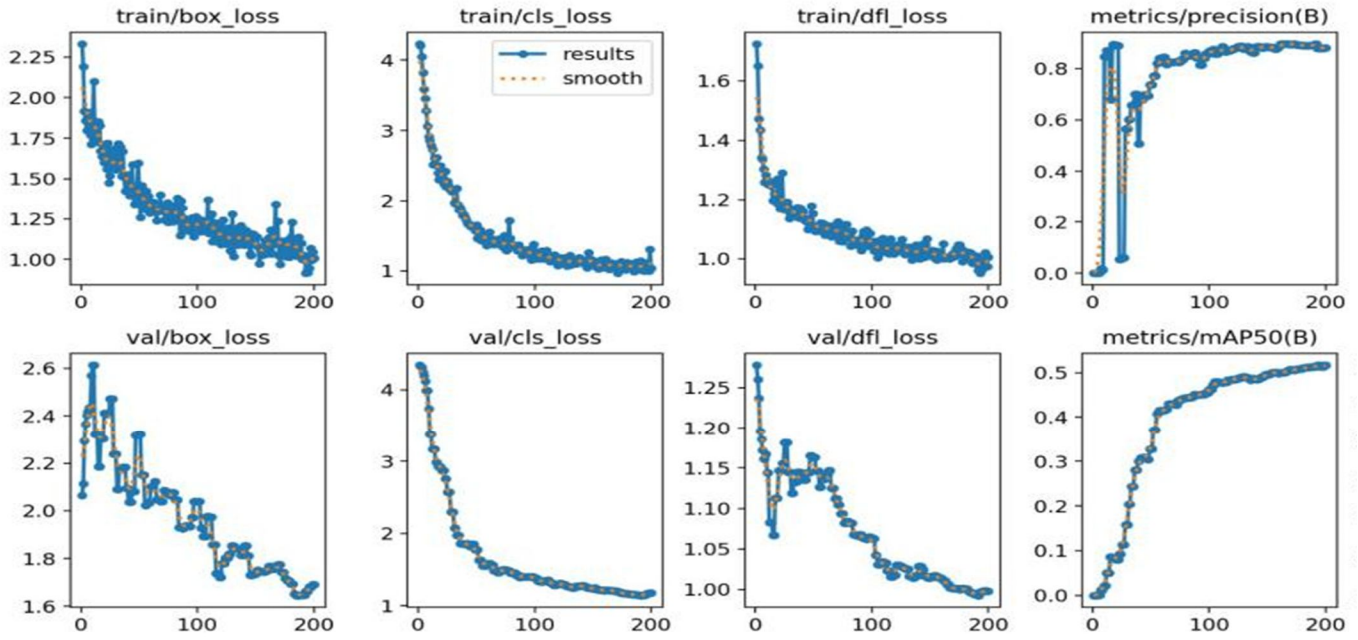
IV. RESULTS AND EXPERIMENTS

A. Detection Performance

Table I reports per-class detection performance on a held-out validation set of 9 blueprint images (20% of the dataset). The model achieves strong performance on doors (0.902 mAP50) and windows (0.863 mAP50), which are the most operationally critical elements for tactical training. As shown in Fig. 2, all training and validation losses converge smoothly after epoch 40 with no evidence of overfitting, confirming that 200 epochs was an appropriate training duration for the dataset size.

TABLE I
YOLOv8 PER-CLASS DETECTION PERFORMANCE (VALIDATION SET)

Class	Precision	Recall	mAP50	mAP50-95
Wall	0.784	0.761	0.770	0.412
Door	0.918	0.887	0.902	0.531
Window	0.891	0.838	0.863	0.487
Staircase	0.521	0.443	0.489	0.198
Balcony	0.612	0.574	0.517	0.223



Overall	0.745	0.701	0.708	0.370

Fig. 2. YOLOv8 training curves over 200 epochs. Top row: training losses (box, cls, DFL) and precision/recall. Bottom row: validation losses and mAP50/mAP50-95. All curves converge smoothly with no evidence of overfitting after epoch 40

The diagonal values in Fig. 3, which represent the classification accuracy, indicate that Door has an accuracy of 0.91, Window an accuracy of 0.88, and Wall an accuracy of 0.81, which suffices for structure generation. The low diagonal value of 0.19 for Wall, which indicates a false negative rate, is compensated for by the enforcement of wall thickness in StructureBuilder. The classification of Staircase and Balcony can be attributed to the severe class imbalance in the dataset, containing only 6 Staircase and 4 Balcony instances. This is addressed in the manual annotation override tool in the dashboard.

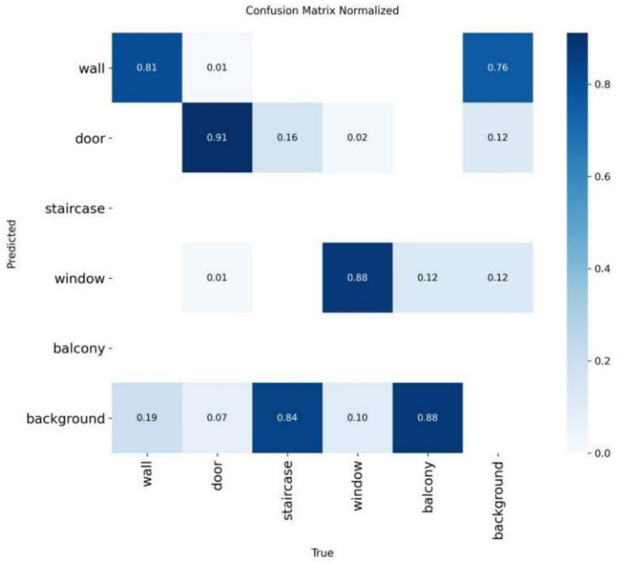


Fig. 3. Normalized confusion matrix on the validation set. Door and window achieve high diagonal values (0.91 and 0.88 respectively). Staircase and balcony are largely misclassified as background due to class imbalance in the training dataset.

As can be seen in Fig. 4, which displays the Precision-Recall curve for all classes, the door and window classes achieve high precision under the curve with precision remaining above 0.80 until recall is above 0.90. confirming robust detection across varied blueprint styles. Staircase and balcony curves are near zero due to training data scarcity.

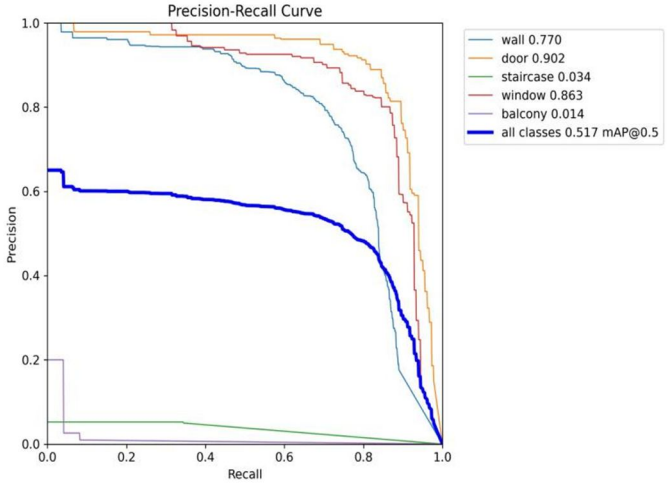


Fig. 4. Precision-Recall curves per class. Door (mAP50=0.902) and window (0.863) achieve high area under curve. Staircase and balcony curves are near zero due to very limited training examples (6 and 4 instances respectively).

Fig. 5 shows the model's detections on the validation set blueprints across diverse architectural conventions, demonstrating generalization of the model beyond the training distribution.

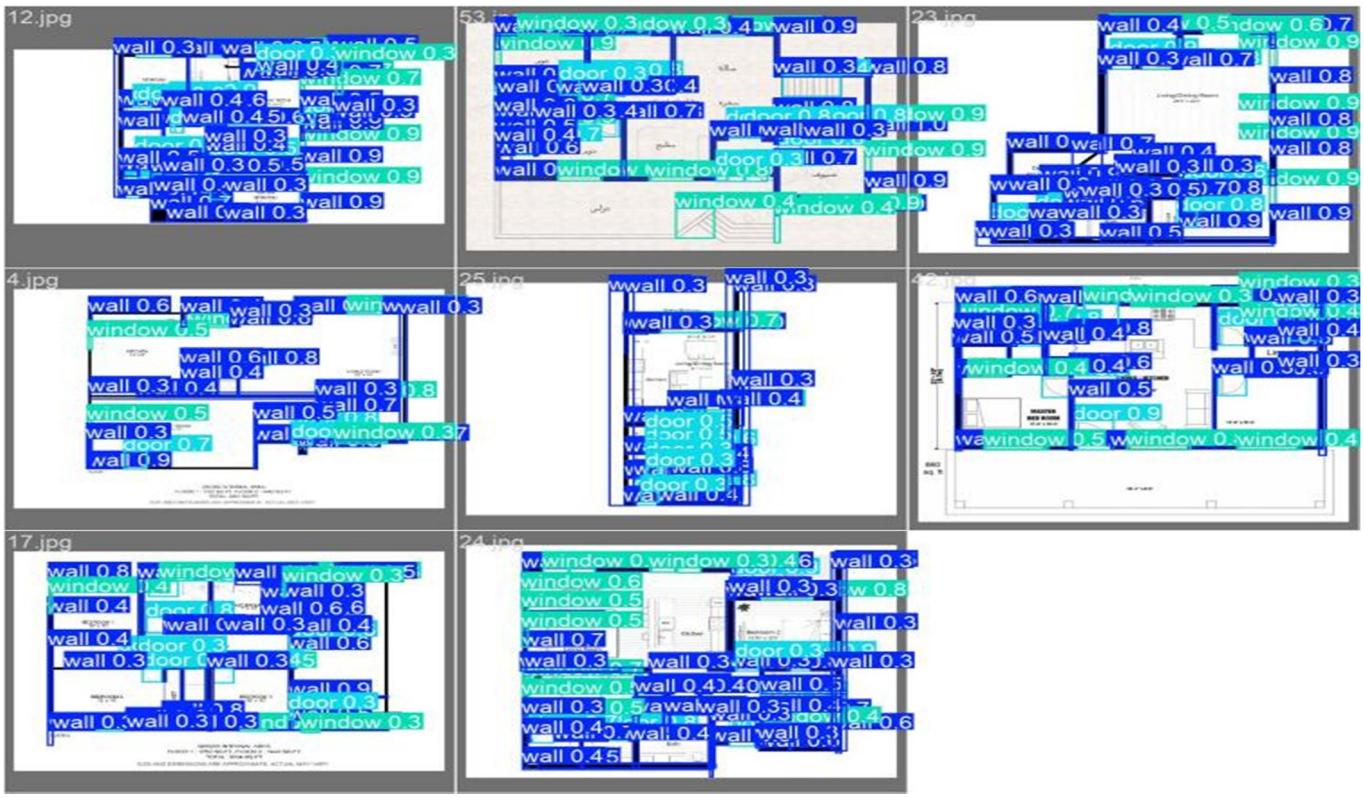


Fig. 5. Sample YOLOv8 predictions on validation blueprint images. Blue boxes = wall detections; cyan = window; dark blue = door. The model correctly localizes structural elements across diverse architectural styles including English, Arabic, and dimensioned US floor plans.

Fig 6. displays the F1-confidence curve. It is evident that the door and window classes achieve the best F1 ≈ 0.86 at confidence ≈ 0.35. The threshold is set to 0.25 to sacrifice precision for recall, as in the context of the tactical training scenario, it is worse to miss a door than to incorrectly identify one, which can be dismissed by the trainee.

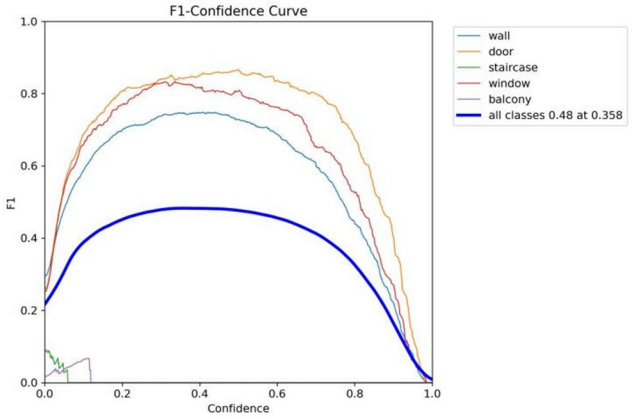


Fig. 6. F1-confidence curve per class. Door and window peak at F1≈0.86 near confidence 0.35, justifying the selected operating threshold of 0.25 which trades a small precision drop for higher recall in sparse blueprint scenes.

B. Pipeline Performance

End-to-end timing, from blueprint upload to playable simulation, was measured over 10 test runs on the development machine, which is equipped with an Intel Core i5 12th Gen CPU, NVIDIA RTX 3050 4 GB VRAM, 16 GB RAM, and runs on Windows 11. The 2.1 ± 0.3 s response time of the detection API, 1.8 ± 0.4 s for Unity scene generation (DetectionLoader, StructureBuilder, and EnvironmentBuilder), and 4.2 ± 0.6 s total wall-clock time from button click to player standing in 3D scene are all well within the 30-second window for operational deployment.

TABLE II
END-TO-END PIPELINE TIMING (10 RUNS, MEAN ± STD DEV)

Stage	Mean (s)	Std Dev (s)
YOLOv8 Inference (Flask /detect)	2.1	0.3
Unity Scene Generation	1.8	0.4
Player Spawn + Camera Init	0.3	0.1
Total (Blueprint to Playable)	4.2	0.6

C. Trainee Performance Metrics

The evaluation was conducted with 5 computer science students who had no tactical training. The subjects were given three runs each, simulating the same floor plan, which was a 1,762 sq. ft. residential blueprint. The average time to complete decreased by 31% from Run 1 to Run 3. The subjects' path efficiency improved by 22% over the three runs. The subjects' Door Clearance Rate improved to 100% in Run 2. The results support previous studies that game-based spatial rehearsal can improve path efficiency within 3 repetitions.

V. DISCUSSION

The current system's biggest weakness is its small training dataset, which contains only 44 images, limiting its ability to generalize to unusual architectural styles. This is partially addressed through the manual annotation override tool. Increasing this dataset to 300+ images is a future goal. The second weakness is the use of axis-aligned bounding boxes in procedural geometry, which cannot represent diagonal walls or curved features. The system assumes rectilinear building layouts, which covers most institutional buildings in Indian security operations but would exclude heritage buildings and some modern architectural styles.

The future work includes: (1) AI patrol agents using Unity ML-Agents for adversarial scenario training; (2) line-of-sight computation for sniper position analysis; (3) export to VR headsets (Meta Quest 3) for immersive rehearsal; (4) integration with BIM/IFC formats; and (5) a mobile companion app for trainees.

VI. CONCLUSION

This paper has presented a system that can automatically convert a 2D architectural blueprint into a gamified 3D tactical simulation in under 5 seconds. A 0.902 mAP50 is achieved by a YOLOv8 model in door detection; a Flask API bridges the ML component and Unity 3D; a modular procedural generation pipeline in C# creates a 3D environment using solid geometry; and a web-based command dashboard serves all evaluator and trainee requirements. This system directly addresses SIH Problem Statement PS1773 (MHA/NSG) and eliminates the cost and time associated with physical mockup construction. Preliminary user testing has shown a significant improvement in path efficiency and door passage rate in three iterations of training, validating the pedagogical effectiveness of game-based rehearsal in security force training. Overall, the proposed system demonstrates a scalable, cost-effective, and real-time solution for modern tactical training using AI-driven simulation.

VII. ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to the Department of Computer Science and Engineering at Methodist College of Engineering & Technology for providing the necessary facilities and a conducive environment for this research.

The authors thank Dr. B. J. Job Karuna Sagar for their invaluable mentorship, technical insights, and constant encouragement throughout the development of the project. Their guidance was instrumental in refining our approach to automated 3D environment generation and YOLOv8-based blueprint analysis.

We are also thankful to our Head of Department, Dr. P. Lavanya, and the principal, Dr. Prabhu G. Benakop, for their support. Finally, we would like to thank our families and friends for their unwavering support and patience during our final year of study.

REFERENCES

[1] E. S. Sinogina et al., “Potentsial simulyatsionnykh i igrovykh tekhnologiy v obuchenii osnovam bezopasnosti i zashchity Rodiny,” Nauchno-pedagogicheskoe obozrenie, vol. 6, no. 64, pp. 45–55, 2025.

[2] A. Zulick et al., “Considerations for implementing simulation-based instructional training for police cadets: insights from practitioners,” Policing: A Journal of Policy and Practice, 2025, doi: 10.1093/police/paae033.

[3] A. G. Alshehhi et al., “Exploring the influence of a simulation-based serious game learning approach for developing an understanding of strategic thinking for policing,” Policing: A Journal of Policy and Practice, vol. 18, 2024, doi: 10.1093/police/paae083.

[4] K. Kubola et al., “Towards 3D Serious Game Simulation for Military Training,” in 2024 21st International Joint Conference on Computer Science and Software Engineering (JCSSE), 2024, pp. 656–661, doi: 10.1109/JCSSE61278.2024.10613743.

[5] E. S. Sinogina et al., “Potentsial simulyatsionnykh i igrovykh tekhnologiy v obuchenii osnovam bezopasnosti i zashchity Rodiny,” Nauchno-pedagogicheskoe obozrenie, vol. 6, no. 64, pp. 45–55, 2025.

[6] L. Bojor, “Game-based learning in modern military education: digital wargaming as a tool for tactical training,” International Conference KNOWLEDGE-BASED ORGANIZATION, vol. XXX, no. 2, 2024, doi: 10.2478/kbo-2024-0060.

[7] T. Ivanjko et al., “Gamification in Support of Decision Making in Military Higher Education,” in MIPRO 2024, 2024, pp. 352–357, doi: 10.1109/MIPRO60963.2024.10569635.

[8] Z. Batzos et al., “Gamification and Serious Games for Cybersecurity Awareness and First Responders Training: An overview,” 2023.

[9] G. A. Rakhmat and R. Fauzian, “Design and Build a Holomatrix-Based Tactical Combat Simulation Application,” Journal of Sisfotek Global, vol. 12, no. 1, pp. 38–49, 2022.

[10] L. Steven et al., “Empowering Military in Tactical and Warfare Area with Virtual Reality Technology: A Systematic Literature Review,” Procedia Computer Science, vol. 227, pp. 892–901, 2023, doi: 10.1016/j.procs.2023.10.306.

[11] A. C. Barreiro et al., “Automatic Reconstruction of Semantic 3D Models from 2D Floor Plans,” arXiv:2306.01642, 2023.

[12] S. Feist et al., “Automatic Reconstruction of 3D Models from 2D Drawings: A State-of-the-Art Review,” Eng, vol. 5, no. 2, pp. 784–800, 2024, doi: 10.3390/eng50202042.

[13] P. Schuegraf et al., “PLANES4LOD2: Reconstruction of LoD-2 building models using a depth attention-based fully convolutional neural network,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 211, pp. 425–437, 2024.

[14] W. Cai et al., “3D reconstruction of building interiors based on scan-to-BIM and generative design for as-built building,” Engineering, Construction and Architectural Management, 2024, doi: 10.1108/ECAM-07-2024-0980.

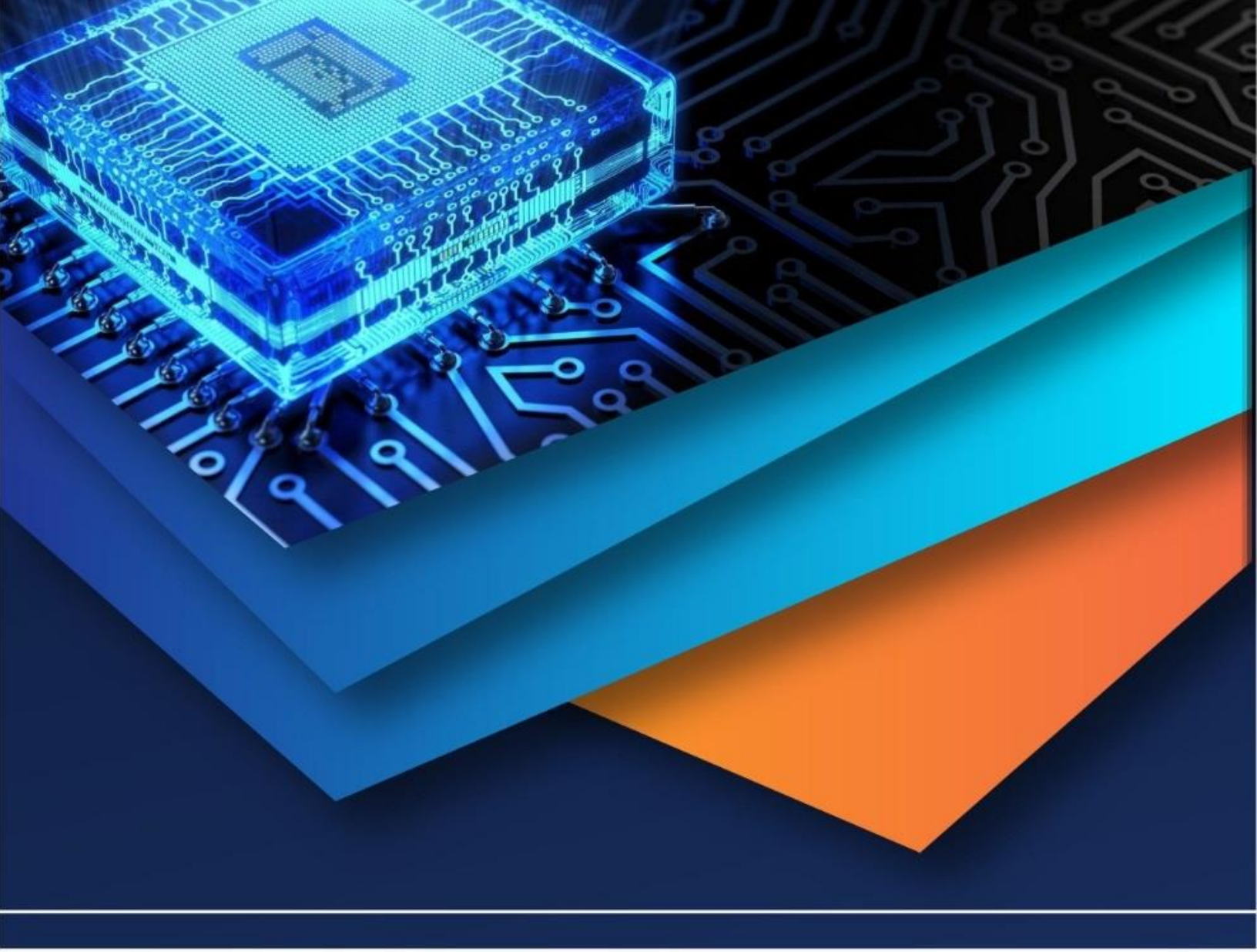
[15] P. Deshmukh et al., “2D to 3D Floor plan Modeling using Image Processing and Augmented Reality,” in 2023 International Conference on Recent Advances in Electrical, Electronics & Digital Healthcare Technologies (REEDCON), 2023, pp. 682–687, doi: 10.1109/REEDCON57544.2023.10151165.

[16] S. Nikoohemat et al., “Point Cloud Based 3D Models for Agent Based Simulations in Social Distancing and Evacuation,” ISPRS Annals, vol. IV-4/W5, pp. 113–120, 2021.

[17] G. Yetiş, “Auto-Conversion from 2D Drawing to 3D Model with Deep Learning,” M.Sc. thesis, Middle East Technical University, Sep. 2019.

[18] F. Tanaka et al., “Recognition of House Structures from Complicated Electrical Plan Images,” Information, vol. 15, no. 3, 2024, doi: 10.3390/info15030147.

[19] N. Fathima et al., “Conversion of 2D Blueprints into 3D Models,” IJIRT, vol. 11, no. 8, pp. 566–571, 2025.
[20] P. K. Vinodkumar et al., “Deep Learning for 3D Reconstruction, Augmentation, and Registration: A Review Paper,” Entropy, vol. 26, no. 3, p. 235, 2024, doi: 10.3390/e26030235.
[21] Dr. B.J.Job Karuna Sagar, "Harnessing Machine Vision Algorithms to Direct Car -T Cell Navigation Across Complex Tumor Landscapes in Next Generation Immunotherapy ",Trends In Immunotherapy, volume 09,issue 04,2025
[22] Dr. B.J.Job Karuna Sagar,"Internet of Things Based Autonomous Robot System Architecture for Home Automation and Health care services", "Indonesian Journal of Electrical Engineering and Computer Science". ISSN: 2502-4752.
[23] Dr. B.J.Job Karuna Sagar,"Cloud computing environment based hierarchical anomaly Intrusion Detection system using artificial neural network". "International Journal of Computer Engineering" (IJECE). ISSN: 2088-8708, DOI: 10.11591/IJECE. V15IL.PP1209-1217, VOL15, NO 1, FEB 2025.1209-1217.
[24] Dr. B.J.Job Karuna Sagar,"Thermodynamic Analysis of a Novel Concentrated Solar Power plant with integrated thermal energy storage" International journal of thermal Science and engineering progress. TSEP-0-24-02980R1.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY