Advances in Artificial Intelligence and Machine Learning; Research 4 (4) 2927-2949 Received 16-10-2024; Accepted 01-12-2024; Published 08-12-2024

AI-Enabled Unmanned Agents for Real-Time 3D Scene Analysis in Military Operations

Giuseppe Macario gm@mit.edu.it

Universitas Mercatorum, Piazza Mattei 10, Rome, Italy

Ministry of Defense, Via XX Settembre 8, Rome, Italy

Corresponding Author: Giuseppe Macario

Abstract

This paper presents an innovative system for dynamic 3D scene perception tailored for battlefield environments, leveraging unmanned intelligent agents equipped with binocular vision and inertial measurement units (IMUs). The system processes binocular video streams and IMU data to deploy advanced deep learning techniques, including instance segmentation and dense optical flow prediction, facilitated by a specially curated target dataset. The integration of a ResNet101+FPN backbone for model training results in a combat unit type recognition accuracy of 91.8%, a mean Intersection over Union (mIoU) of 0.808, and a mean Average Precision (mAP) of 0.606. A dynamic scene localization and perception module utilizes these deep learning outputs to refine pose estimations and enhance localization accuracy by overcoming environmental complexities and motion-induced errors, typically associated with SLAM methodologies.

Application tests conducted within a simulated battlefield metaverse environment demonstrate a 44.2% improvement in self-localization accuracy over traditional ORB-SLAM2 Stereo methods. The system efficiently tracks and annotates dynamic and static battlefield elements, continuously updating global maps with precise data on agent poses and target movements. This work not only addresses the dynamic complexity and potential information loss in battlefield scenarios but also lays a foundational framework for future enhancements in network capabilities and environment reconstruction methodologies. Future developments will focus on precise identification of combat unit models, multi-agent collaboration, and the application of 3D scene perception to advance real-time decision-making and tactical planning in joint combat scenarios. This approach holds significant potential for enriching the battlefield metaverse, fostering deep human-machine interaction, and guiding practical military applications.

Keywords: Metaverse, Virtual Reality, Military Technology, Artificial Intelligence, Machine Learning, Robotics, Motion Tracking, 3D Reconstruction, Image Segmentation, Optical Flow.

1. INTRODUCTION

As military equipment continuously evolves and battlefield environments become increasingly complex, modern warfare is characterized by the expanding scale of battlefields and a higher reliance on information technology. Efficient and accurate three-dimensional (3D) perception of battlefield environments is crucial during both preparation and operational phases of warfare. This perception enables the acquisition of enemy deployment information, assisting commanders in analyzing the battlefield situation in real time, assessing threat levels, and making optimal command and control decisions.

The development of unmanned systems and the need for enhanced battlefield perception have led to drones and smart detection vehicles increasingly taking on tactical reconnaissance and information support roles, becoming central to battlefield operations. In combat, it is vital to obtain real-time, accurate information about the enemy's main combat units, including types, movements, and 3D positions. This information must be continuously updated and transmitted to combat personnel and command centers, facilitating the analysis of enemy deployments and the formulation of targeted countermeasures.

With the advancement of unmanned intelligent agents and diverse sensors, 3D scene perception inspired by bionic vision has emerged as a significant area of research [1, 2]. Although visual-based methods for positioning and target recognition are rapidly evolving, few are suitable for the complex conditions of battlefield environments, where dynamics, target concealment, and occlusions can impair the accuracy of target recognition.

Furthermore, the "battlefield metaverse", supported by simulation and virtual reality technologies, provides highly realistic, immersive, and integrated combat scenarios. This metaverse extensively simulates military operations and decision-making processes, offering accurate field data and high experimental efficiency with significant practical implications. It offers detailed reconstructions of real-world scenes, models of combat units, and data on battlefield confrontations, thereby supporting decision-making and training in command centers.

This paper proposes a system that leverages sensor data from unmanned intelligent agents to achieve robust environmental and target perception, thereby enhancing battlefield situational awareness. Utilizing binocular vision and an Inertial Measurement Unit (IMU), coupled with deep learning networks and dynamic simultaneous localization and mapping (SLAM) technology, this system aims to achieve dynamic 3D scene perception of the battlefield based on operational data from the battlefield metaverse.

2. SYSTEM REQUIREMENTS AND TECHNICAL SOLUTION

2.1 Functional Requirements

Unmanned intelligent agents equipped with binocular vision devices and Inertial Measurement Units (IMUs) are crucial for perceiving and reconnoitering in battlefield environments. These agents swiftly traverse the battlefield, continuously collecting video streams along with tri-axial translational and rotational speed and acceleration data. This data is transmitted back to a control terminal for real-time processing and analysis. The system is designed to fulfill two primary tasks: 1. achieve self-localization by determining the unmanned agent's position through sensor fusion and sparse environmental reconstruction, 2. automatically detect battlefield targets based on the agent's location, transforming the localization data from the camera's coordinate system to the global battlefield coordinate system, thereby identifying the type, movement, and position of targets.

The agents must account for variable battlefield terrain, which includes movements such as pitch, roll, and yaw. Accurately acquiring attitude information is essential for precise target localization and conversion. The processed data assists commanders in situational assessment and decision-making.

2.2 Technical Challenges

The dynamic and complex nature of battlefield environments presents significant challenges to self-localization and target detection technologies [3]. Techniques such as Simultaneous Localization and Mapping (SLAM) [3–9], exemplified by ORB-SLAM2 [10], can facilitate sparse environmental reconstruction and self-localization. Additionally, Convolutional Neural Networks (CNNs) and other deep learning models are proficient in processing image data. However, these methods have intrinsic limitations.

SLAM techniques, which typically assume environmental rigidity and static objects, struggle in battlefields characterized by moving targets and repetitive textures. This can lead to drift or loss of localization due to disrupted image feature extraction and pose estimation. High-mobility unmanned agents, which maneuver to evade attacks or execute tactical movements, frequently lose localization during rapid maneuvers or sharp turns if relying solely on visual tracking.

Moreover, battlefield targets often feature mutual occlusion, camouflage, and complex backgrounds of lighting and texture, all of which challenge the accuracy and reliability of target detection [11, 12]. The demands for real-time situational awareness in technologically advanced and strategic battlefield operations are rigorous. Standalone SLAM or target detection algorithms are insufficient for meeting the needs of simultaneous localization and scene perception, revealing a considerable gap in practical applications.

Consequently, there is a need for the enhancement and holistic integration of models and algorithmic modules to address the disturbances presented by dynamic and complex battlefield environments. This approach aims to continually output and update battlefield target information during the operation of unmanned intelligent agents.

2.3 Technical Solution

The rapid development of the battlefield metaverse, which refines the construction of digital battlefield spaces for real-time situation awareness and interaction, has revolutionized traditional training analysis and technical verification methods. This transformation garners widespread attention and application by maximally simulating real-world military operations, scenarios, and decision-making processes. It effectively overcomes the challenges in constructing virtual-real combat systems and conducting technical verifications, thereby holding substantial military value. By integrating dynamic virtual environment data, the metaverse reproduces battlefield terrains, weather conditions, and enemy deployments through software, creating an expansive, complex, and interactive virtual environment. This capability enables rapid adjustments of scenarios and parameters during combat simulations to test combat concepts and new military technologies, facilitating the comprehensive inversion of spatiotemporal battlefield situations and supporting real-time dynamic joint command and control.

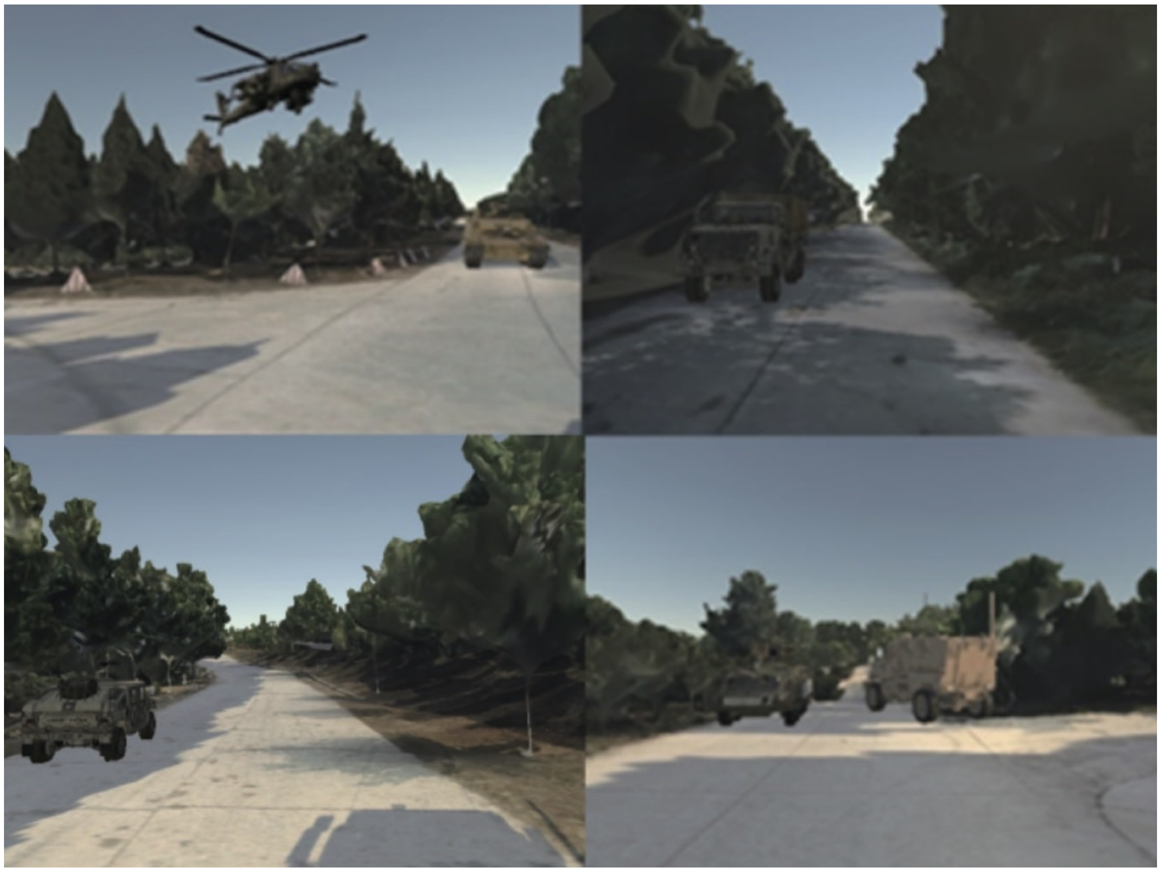


Figure 1: Scenes in the battlefield metaverse

The battlefield metaverse serves as an essential data source, framework for testing, and application platform for the system. Within this digital environment (see FIGURE 1) diverse conditions such as different backgrounds, lighting, and weather are simulated, and models of typical battlefield combat units are integrated. This setup aids in the comprehensive collection, annotation, and creation of a battlefield target dataset, which is crucial for testing and refining the target extraction module. The dataset encompasses details like combat unit types, instance masks, and bounding boxes, providing a rich data source for training deep learning models used in battlefield target extraction. Additionally, unmanned intelligent agents deployed by the command terminal for reconnaissance simulate real landscapes and environments, transmitting collected data back for computing 3D scene perception information to support situation generation and evaluation. The ground truth data from the battlefield metaverse are utilized to assess the accuracy of system outputs.

In response to the demands for scene perception and deployment under complex and dynamic battlefield conditions, the technical solution presented in this paper comprises three main components: raw data processing input, deep learning processing, and dynamic scene localization perception, as shown in FIGURE 2.

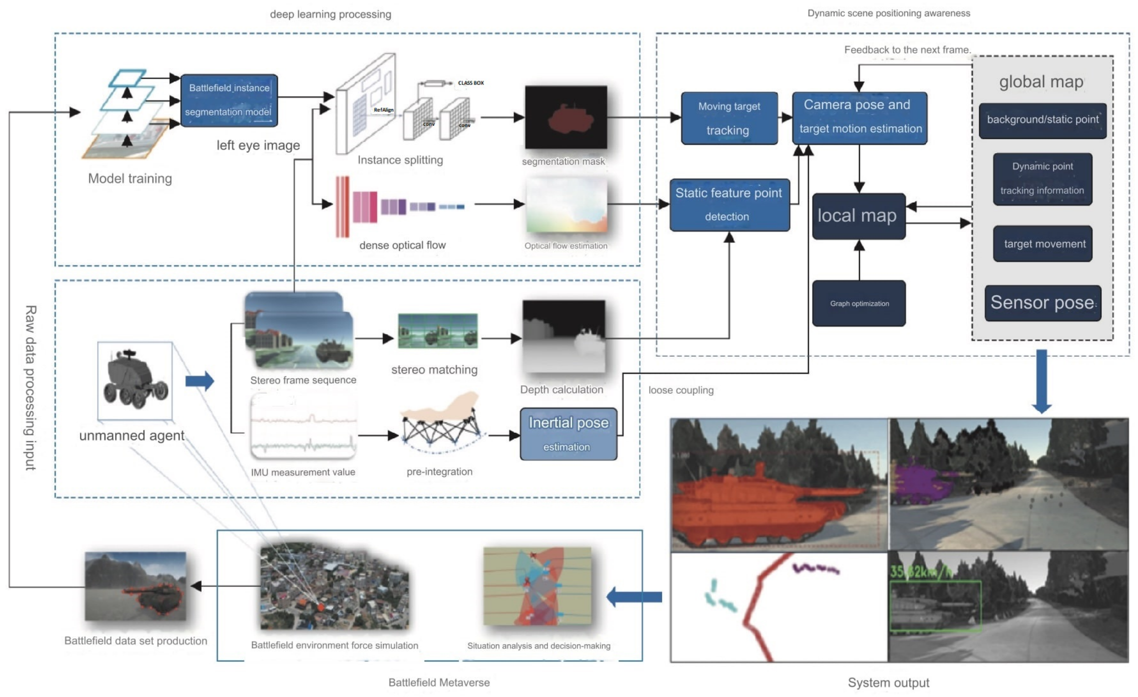


Figure 2: 3D scene perception technical solution

2.3.1 Raw data processing

Unmanned intelligent agents equipped with binocular vision devices continuously capture battlefield reconnaissance video streams at a specified frame rate. These devices, mimicking human eyes, calculate matching points through semi-global stereo matching. Given the known baseline distance between cameras, 3D depth of the scene is reconstructed from 2D images. Concurrently, the inertial measurement unit (IMU) onboard provides tri-axial translational and rotational speed and acceleration data. By pre-integrating this IMU data, pose estimation is facilitated during rapid movements, which is later synchronized with visual information for enhanced localization. The integration of binocular vision and IMU devices, while compact and straightforward in setup, provides extensive scene information crucial for the entire situational awareness system.

2.3.2 Deep learning module

The target detection method identifies the positions and categories of objects within images. Additionally, instance segmentation techniques provide 2D pixel masks of targets, enhancing accuracy in localization while mitigating the impact of dynamic targets on self-localization during SLAM feature point extraction. Deep learning networks are employed to classify, detect, and segment targets. To continually enhance model performance, network depth is increased, which introduces problems of degradation. The implementation of residual networks (ResNet) and feature pyramid networks (FPN) effectively resolves issues related to degradation and multi-scale transformations, thereby enriching the semantic information of the models. Consequently, the Mask R-CNN framework, which integrates ResNet and FPN, excels in detecting objects and producing high-quality segmentation masks for each instance. The battlefield target dataset, enriched with augmented data, trains the model to perform precise battlefield target extraction. Furthermore, to ensure robust dynamic target tracking, pixel motion information is integrated, utilizing PWC-NET to estimate dense optical flow between video frames, which captures the motion speed and direction of each pixel. The combination of target extraction and optical flow estimation provides essential scene priors for the system.

2.3.3 Dynamic scene localization and perception

These scene priors are input into the tracking thread, merged with sensor data to estimate the unmanned agent's pose and acquire target localization and movement information. The system dynamically updates a local map for each frame of the video stream using a sliding time window, enhancing the accuracy of the agent's own pose and maintaining a global map throughout the perception process. This method outputs a sparse reconstruction of the static background point cloud, the agent's trajectory, and detailed information on target type and location. This comprehensive situational data is continuously fed back to combat personnel, command terminals, and utilized in modules for battlefield metaverse situational fusion analysis and operational decision-making.

Furthermore, the battlefield metaverse serves as a hybrid virtual-real military ecosystem, offering precise landscape and combat unit data sources that facilitate efficient system construction and testing. The command terminal deploys unmanned agents within this simulated environment to closely mimic real combat scenarios. While the system algorithms focus on 3D scene perception in dynamic and complex environments, they do not extend to discussing the complete architecture technology of the metaverse.

3. DEEP LEARNING SCENE PRIORS

3.1 Battlefield Target Instance Segmentation

The battlefield environment encompasses major combat units such as troop carriers, armored vehicles, helicopters, tanks, and UAVs, characterized by complex contours and camouflage that may overlap during battlefield maneuvers. Accurate extraction of target positions, classification, and output of instance masks are essential for generating situational awareness and tracking moving targets.

To achieve effective battlefield target instance segmentation using deep learning, it is crucial to prepare a comprehensive target dataset prior to network training. In the battlefield metaverse, various combat scenarios and unit models are utilized to collect target image data, forming a robust and accurately annotated dataset. The data collection yielded 3956 image sets for eight primary combat units, with detailed naming and type information provided in TABLE 1.

Table 1: Combat units in the battlefield metaverse

Instance	Type	Camouflaged
APC#1	Transport vehicle	No
APC#2	Transport vehicle	Yes
Tank#1	Tank	No
Tank#2	Tank	Yes
ML	Missile launcher	Yes
Hel	Helicopter	No
UAV	UAV	No
Jeep	Jeep	Yes
MRAPV	MRAP vehicle	Yes
Truck	Truck	Yes

Given the diversity of battlefield terrains, the textural features of identical targets vary under different backgrounds and lighting conditions [11, 12]. Moreover, maneuvers such as turret rotation in tanks add to the complexity of feature extraction. Traditional image augmentation techniques like stretching, mirroring, and rotation, along with specific methods employed during data collection, enhance the model's learning capabilities, as shown in FIGURE 3:

Multi-Angle Sampling Captures targets from various angles to ensure comprehensive feature representation.

Camouflage and Partial Occlusion Sampling Includes samples with camouflage patterns and partial occlusions to simulate battlefield conditions.

Lighting Condition Variation Sampling Gathers images under varied lighting conditions to boost model robustness.

Target Maneuver Action Variation Sampling Records targets performing different maneuvers, such as turret rotations for tanks, to capture dynamic battlefield actions.

These methods collectively enrich the dataset ensuring the trained model can handle complex battlefield scenarios and accurately segment targets in real-time applications.

The training of the instance segmentation model is supervised, requiring precise mask ground truth. Within the battlefield metaverse, each target's mask image is exported during data collection, facilitating rapid and bulk generation of Common Objects in Context (COCO) .json files with annotations for each image set. The process consists of the following steps:

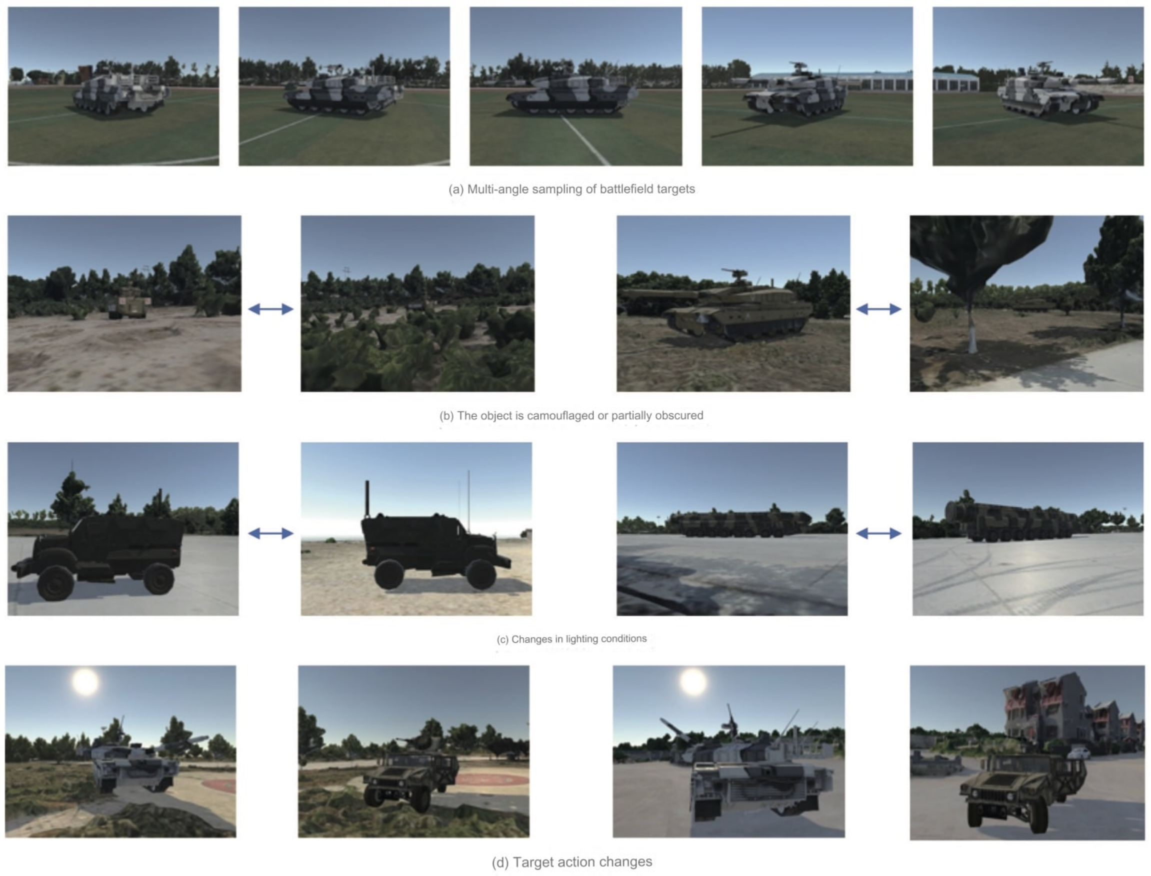
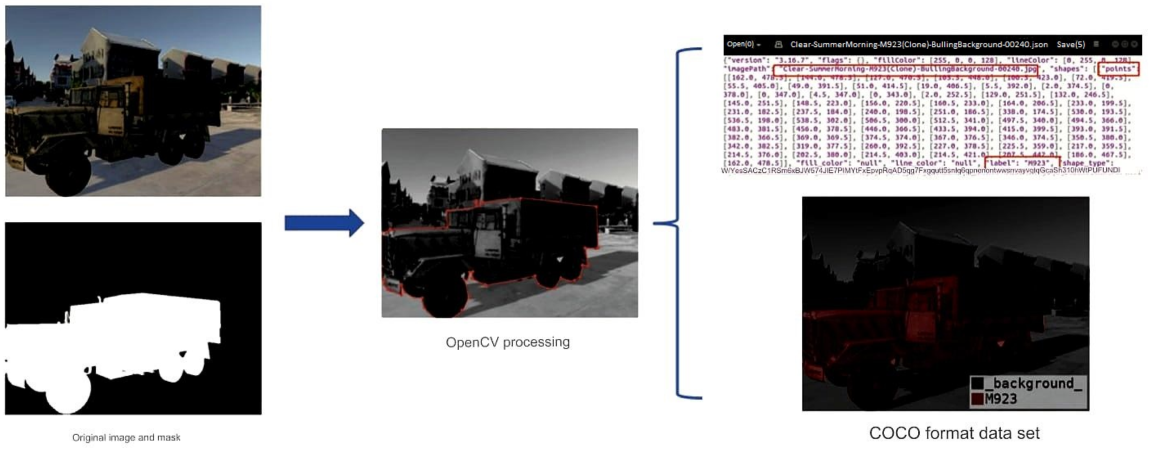

(a) Multi-angle sampling of battlefield targets
(b) The object is camouflaged or partially obscured
(c) Changes in lighting conditions
(d) Target action changes

Figure 3: Image augmentation techniques

1. Use the `cv2.imread()` function to read the input file, obtaining an image represented as a numpy array.
2. Convert the image to grayscale and binarize it using the `cv2.threshold()` function.
3. Extract contour information from the binary image using the `cv2.findContours()` function.
4. Convert each contour to COCO format and add it to a dictionary.
5. Return the dictionary.
6. Repeat the above steps for new image-mask pairs to obtain the .json files.

The data is organized into a format recognized by the model, as shown in FIGURE 4.

The Mask R-CNN network, utilizing a ResNet-101 backbone combined with a feature pyramid network (FPN), is employed to enhance feature representation capability. This high-precision, highly adaptable network includes modules for image feature extraction, region proposal generation, and object detection, providing outputs for classification, bounding box prediction, and instance masks through fully connected layers.



```
{"version": "3.16.7", "flags": {}, "fillColor": [255, 0, 0, 128], "lineColor": [0, 255, 0, 128],
"imagePath": "Clear-SummerMorning-M923(Clone)-BullingBackground-00240.jpg", "shapes": [{"points":
[[162.0, 478.5], [144.0, 478.5], [127.0, 476.5], [103.5, 448.0], [100.5, 423.0], [72.0, 419.5],
[55.5, 405.0], [49.0, 391.5], [51.0, 414.5], [19.0, 406.5], [5.5, 392.0], [2.0, 374.5], [0,
378.0], [0, 347.0], [4.5, 347.0], [0, 343.0], [2.0, 252.5], [129.0, 251.5], [132.0, 246.5],
[145.0, 251.5], [148.5, 223.0], [156.0, 220.5], [160.5, 233.0], [164.0, 206.5], [233.0, 199.5],
[231.0, 182.5], [237.5, 184.0], [240.0, 198.5], [251.0, 186.5], [338.0, 174.5], [530.0, 193.5],
[536.5, 198.0], [538.5, 302.0], [506.5, 300.0], [512.5, 341.0], [497.5, 340.0], [494.5, 366.0],
[483.0, 381.5], [456.0, 378.5], [446.0, 366.5], [433.5, 394.0], [415.0, 399.5], [393.0, 391.5],
[382.0, 366.5], [369.0, 369.5], [374.5, 374.0], [367.0, 376.5], [346.0, 374.5], [350.5, 380.0],
[342.0, 382.5], [319.0, 377.5], [260.0, 392.5], [227.0, 378.5], [225.5, 359.0], [217.0, 359.5],
[214.5, 376.0], [202.5, 380.0], [214.5, 403.0], [214.5, 421.0], [207.5, 442.0], [186.0, 467.5],
[162.0, 478.5]], "fill_color": "null", "line_color": "null", "label": "M923", "shape_type":
W/YesSACzC1RSm6xBJW574JIE7PIMYtFxEpvpRqAD5qg7Fxgqutt5snlq6qpnononttwwsnvayvqlqGcaSh310hWtPUFUNDI
```

Figure 4: Dataset creation

The model is trained on a server equipped with an NVIDIA A2 GPU running Ubuntu 24.04 LTS, using TensorFlow 2.16.1. Key training parameters such as learning rate and training epochs are optimized based on the results from the training and validation sets.

The total loss function of the model, combining classification loss L_{class}, bounding box regression loss L_{box}, and average binary cross-entropy loss L_{mask}, is defined as:

$$L = L_{\text{class}} + L_{\text{box}} + L_{\text{mask}} \tag{1}$$

This function evaluates the discrepancy between predicted and actual object categories, bounding box positions, and segmentation masks. The loss curves demonstrate a rapid decrease after the onset of training, eventually stabilizing as the model adapts to the features of the training set, as shown in FIGURE 5.

Validation on the test set confirms the model's capability to accurately detect and segment various target types, even under challenging conditions such as partial occlusion, low light, and overlapping instances, as shown in FIGURE 6.

FIGURE 7 shows the model's Intersection over Union (IoU) curve. IoU represents the ratio of the intersection to the union of the predicted mask and the ground truth mask on the test set, reflecting the effectiveness of instance segmentation. Experiments on the test set indicate that the model's accuracy for recognizing types of combat units is 91.8%, with a maximum IoU of 0.937 and a minimum not lower than 0.643, and an average IoU of 0.808. For larger targets with more regular edges, such as trucks, the model tends to obtain better masks, while for targets like helicopters, which have many irregular or overlapping edges, the model may misclassify pixels in blurred areas, resulting in a lower IoU. By calculating recall and precision, the model achieves a mean Average Precision of 0.604. The test results show that the model maintains a high average IoU, and the predicted mask results closely fit the ground truth.

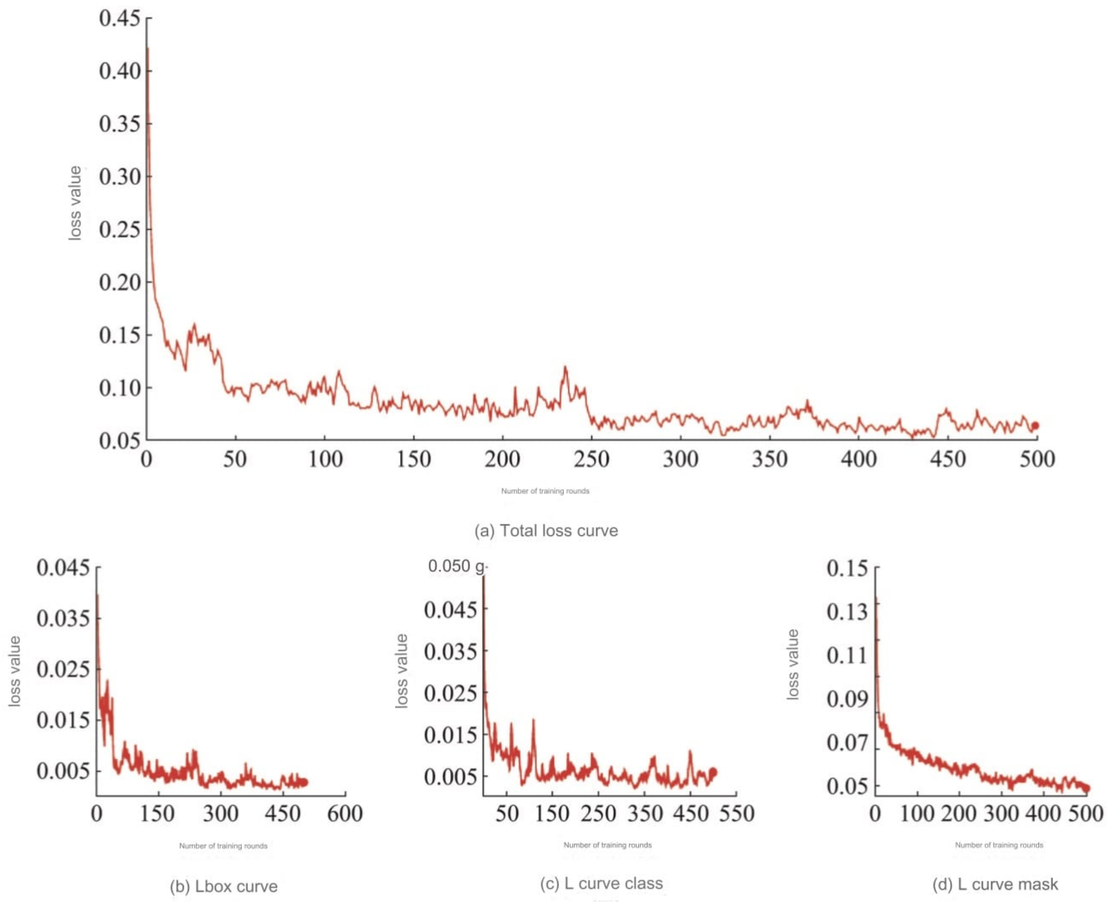


Figure 5: Model training loss curve

The output file of instance segmentation is a matrix of the same size as the image, with the background labeled as 0, and each detected instance is labeled with a different tag. This can provide target information in the scene for the subsequent SLAM process, as shown in FIGURE 8.

3.2 Dense Optical Flow Calculation

The motion of a moving target in a scene can be characterized using motion equations. For images captured by visual devices, changes in pixel brightness can represent this motion, constructing an optical flow field to map three-dimensional motion vectors onto the two-dimensional camera plane. Both the translation and rotation of the visual device itself, as well as the movement of objects within the field of view, generate optical flow vectors on the imaging plane. By calculating the optical flow vector for each pixel, a dense two-dimensional optical flow field is formed. The energy function of the dense optical flow is minimized to obtain the relative pixel displacement (u, v). The integral

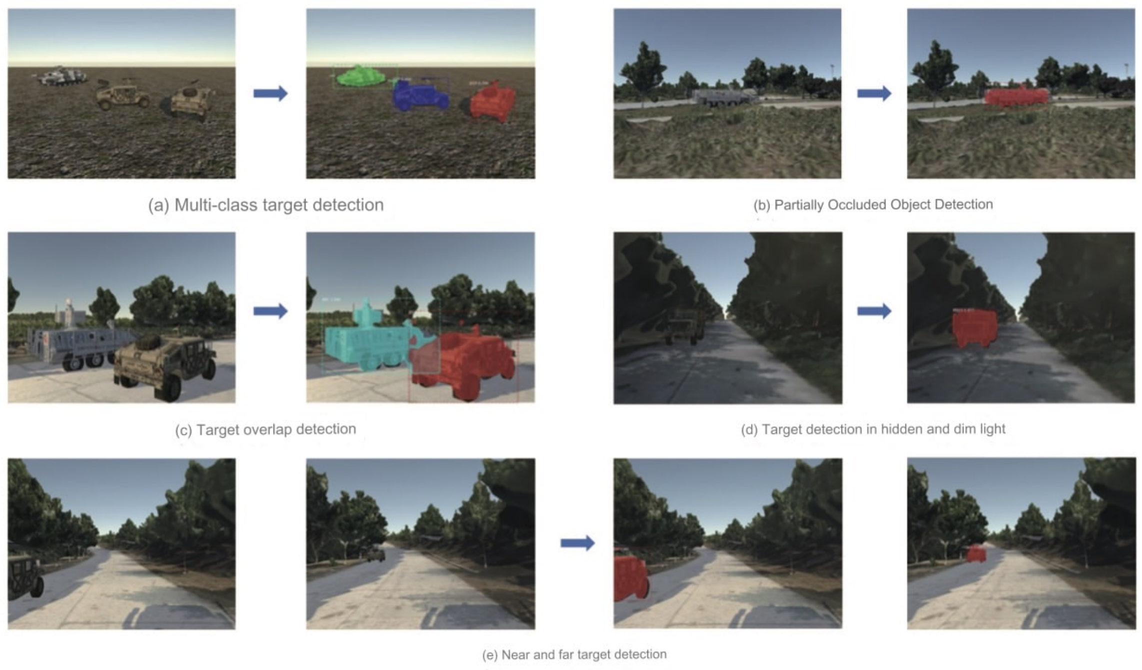

(a) Multi-class target detection
(b) Partially Occluded Object Detection
(c) Target overlap detection
(d) Target detection in hidden and dim light
(e) Near and far target detection

Figure 6: Battlefield combat unit segmentation test

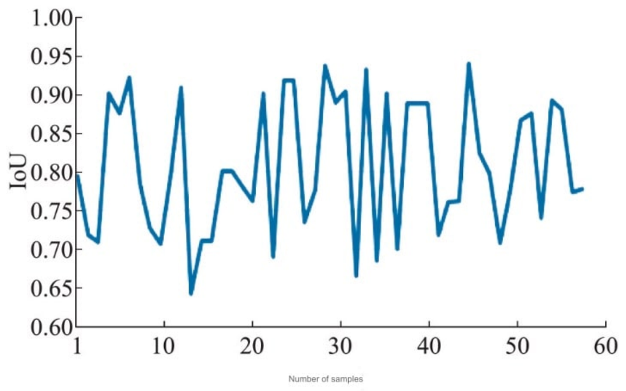


Figure 7: Instance segmentation IoU curve

energy function is described as:

$$\min_{u,v} E(u,v) = \iint [\varphi(T(x,y) - I(x+u, y+v)) + \lambda \cdot \psi(|\nabla u|, |\nabla v|)]\, dx\, dy$$

where φ is the error function, T represents the previous frame image, I is the current frame image, and $\lambda \cdot \psi(|\nabla u|, |\nabla v|)$ acts as the smoothness term, ensuring the displacement is consistent near strong feature points in areas with distinct textures. ψ is also an error function used to calculate the flow vector (u, v), reflecting the extent of pixel motion.

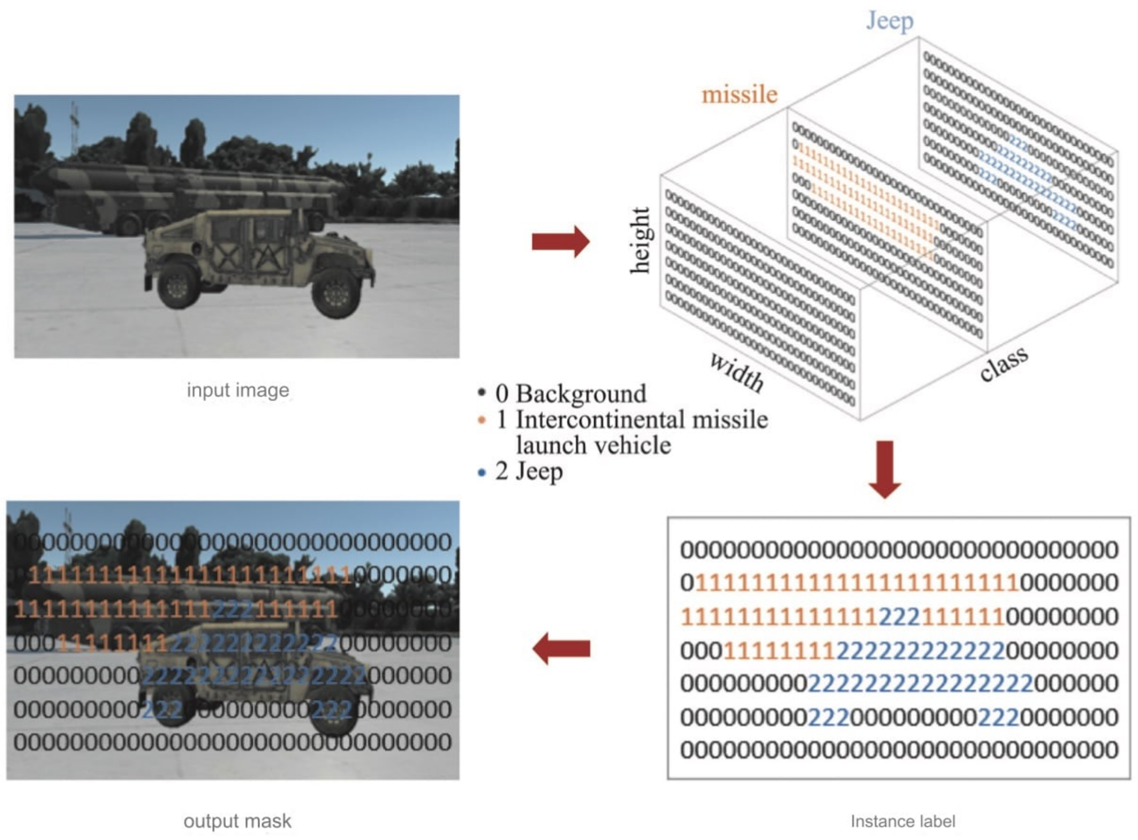


Figure 8: Instance segmentation module output

Using CNNs to estimate the optical flow field enables precise calculation of optical flow vectors for each pixel during inter-frame motion, essential for moving target detection and tracking. The CNN-based PWC-NET network employs feature pyramids at its input layer to enable projection and estimation of optical flow at various scales. Each level of the pyramid uses convolutional neural networks to estimate pixel-level optical flow and incorporates a pyramid optical flow refinement layer, which combines the estimated results with input image features to achieve more accurate and less noisy optical flow estimation.

PWC-NET, implemented using TensorFlow, follows a methodology documented in relevant literature. It is initially trained on the FlyingChairs dataset [13, 14], which includes optical flow image pairs with precise annotations and high-quality images. Further fine-tuning occurs on the FlyingThings3D dataset [15], with additional refinements made using the KITTI tracking dataset [16] that features multi-vehicle road scenes. This enhances the model's ability to adapt to diverse motion scenarios and improve optical flow estimation under varying lighting conditions, as shown in FIGURE 9.

The network outputs optical flow vectors for each pixel between two images, indicating motion, direction, and speed. In the visualized optical flow images, the direction of the optical flow vector

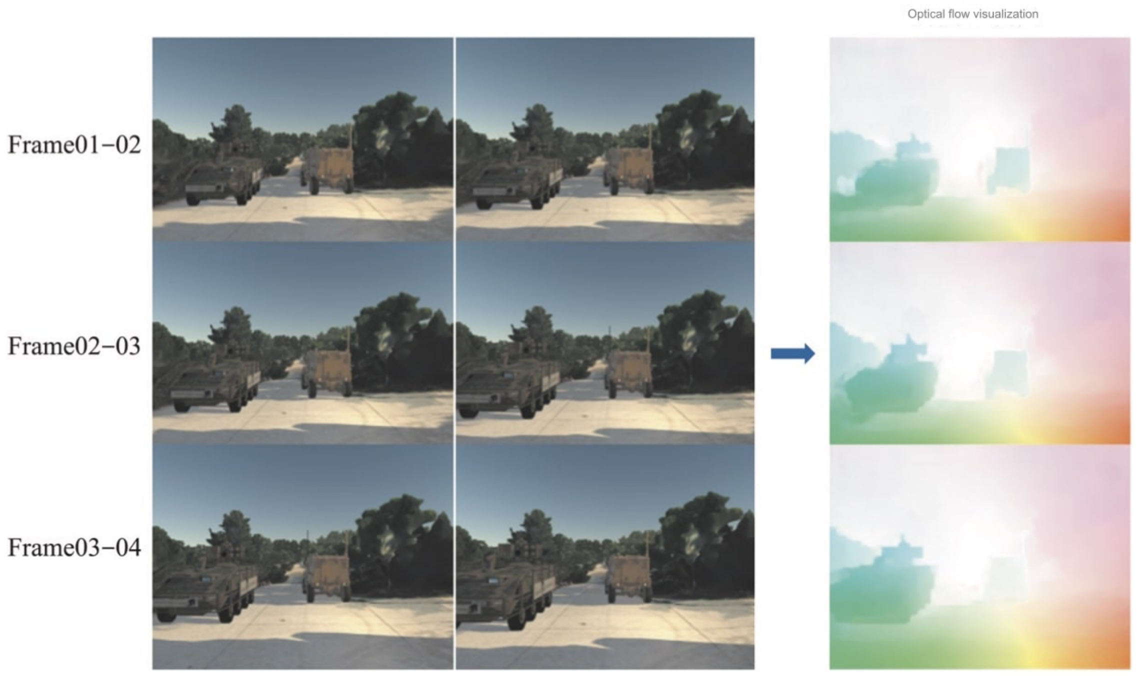


Figure 9: Scene dense optical flow calculation

is encoded using a color wheel where, for instance, red indicates rightward motion and yellow downward. The brightness and intensity of the color represent the speed, with higher color saturation denoting faster movement.

Trained on precise annotation datasets and fine-tuned for specific motion scenarios, the model demonstrates generalized capability for battlefield environments. During unmanned intelligent agent maneuvers, it calculates inter-frame optical flow estimations and converts them into standard .flo files, which can be directly processed by OpenCV for use in localization and perception tasks.

4. LOCALIZATION AND PERCEPTION IN DYNAMIC SCENES

Extracting battlefield target information and dense pixel-level optical flow motion information between frames can serve as important scene priors for the localization and perception process. This helps overcome the constraints imposed by the rigid environment assumption in the SLAM process, allowing for more accurate pose estimation of unmanned agents and enabling the calculation of battlefield target motion based on this information.

By leveraging ORB SLAM2's feature extraction and pose estimation, along with the Visual Dynamic Object-aware method for dynamic object tracking and joint optimization, a dynamic localization and perception module for battlefield environments is constructed. The structure and data flow of the module are shown in FIGURE 10. After the extraction of target priors, a combined tracking and back-end optimization process is conducted for both static and dynamic elements.

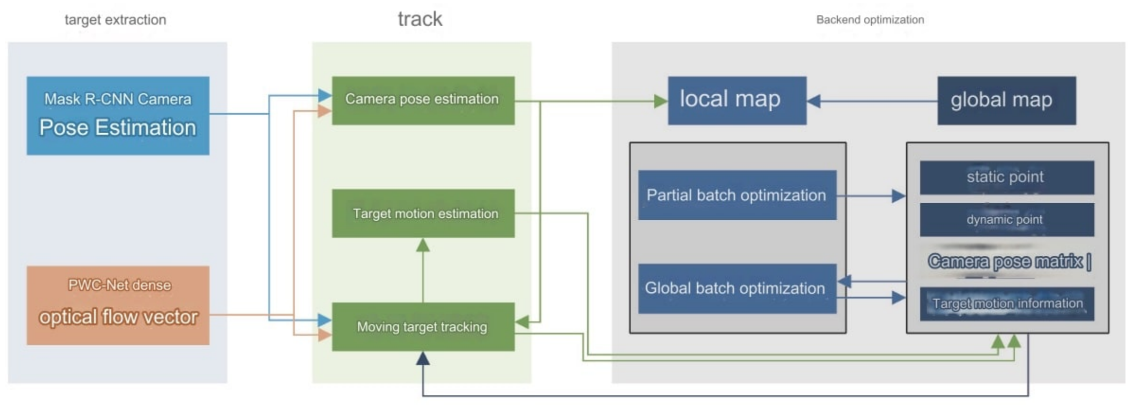


Figure 10: Module structure and data flow

4.1 Pose Estimation and Tracking

The tracking phase involves three concurrent processes: ORB feature point extraction, camera pose estimation, and dynamic target motion estimation. ORB feature point extraction follows the ORB-SLAM2 protocol [10], detecting corner features in each frame and extracting them to form a sparse map output. This method combines FAST (Features from Accelerated Segment Test) keypoint detection and BRIEF (Binary Robust Independent Elementary Features) descriptors to smooth noisy pixel blocks. Instance segmentation provides masks for potential moving targets in frames, allowing for the extraction of ORB features from static parts of the image, excluding dynamic object segments.

After isolating feature points and separating dynamic from static features, static points are used to estimate the pose of the unmanned intelligent agent's left camera. The pose is initially estimated using a combination of the PnP (Perspective-n-Point) and RANSAC (Random Sample Consensus) algorithms, followed by minimizing the reprojection error function, expressed as:

$$e_i(X_k) = \tilde{C}_k - \pi(X_k^{-1} \cdot F_{k-1}) \tag{2}$$

where $\tilde{C}_k$ represents the projection of points from the previous frame F_{k-1} onto frame k, and π is the projection function:

$$\pi(m_k) = Km_k \tag{3}$$

Here, K is the 3x3 intrinsic camera matrix, containing focal length, principal point coordinates, and skew factor, and m_k is a 3D point in the global coordinate system. Using Lie algebra parameterization of the 3D special Euclidean group SE(3), a least-squares error function is established and minimized using the Levenberg-Marquardt algorithm to determine the camera pose matrix X_k.

To improve robustness in dynamic scenes and during high-speed rotations, dense optical flow Φ_k is integrated into the optimization of the pose estimation:

$$e_i(X_k) = C_{k-1} + \Phi_k - \pi(X_k^{-1} \cdot F_{k-1}) \tag{4}$$

Here, Φ_k represents the inter-frame motion vector, enhancing the accuracy of the pose estimation. $\Phi_k = \tilde{C}_k - C_{k-1}$, where C_{k-1} is the projection of static feature points F_{k-1} from frame $k-1$ in the image coordinate system.

Additionally, the system integrates IMU data, which provides tri-axial displacement and rotational acceleration, to enhance visual inter-frame motion estimation. Given the higher frequency of IMU data compared to visual frames, visual odometry employs a loosely coupled fusion of IMU and visual information, estimating poses separately and then aligning them for a comprehensive camera pose.

For dynamic target motion estimation, feature points from target regions identified by Mask R-CNN are used. Let the motion transformation matrix from frame $k-1$ to k be ${}^kO_{k-1}$, and let the dynamic feature points be F_k^D; the motion estimation is

$$F_k^D = {}^kO_{k-1} \cdot F_{k-1}^D \tag{5}$$

The motion transformation matrix from one frame to the next is therefore estimated using reprojection errors constructed around these dynamic feature points, enhanced by incorporating optical flow data:

$$e_i({}^kO_{k-1}) = C_{k-1} + \Phi_k - \pi[X_k^{-1} \cdot ({}^kO_{k-1}F_{k-1}^D)] \tag{6}$$

Using this data, the displacement vector of the target pose matrix is determined, facilitating accurate 3D localization of battlefield targets.

Scene flow, integrating binocular depth and optical flow information, calculates the 3D scene dynamics by evaluating spatial displacements relative to the camera. This data helps identify if a structure within the image field of view is a moving feature. The scene flow of 3D points between frames $k-1$ and k is

$$f_k = F_{k-1} - F_k \tag{7}$$

Scene flow is related only to the scene's motion; any static 3D feature point has zero scene flow. Thus, it directly determines whether a structure in the image field of view is a moving feature. If more than 50% of pixel points in an instance mask area exceed the scene flow threshold of 0.15, the target is classified as moving. The centroid pose transformation of these moving targets is then calculated to obtain their motion velocity in the global coordinate system, further enhancing situational awareness.

4.2 Back-end Optimization and Mapping

The backend of the system is modeled as a graph optimization problem, where nodes represent variables to be optimized, and edges symbolize constraints between these variables. The objective is to derive an optimal solution that adheres to these constraints, thereby refining the camera pose and target localization. The nonlinear optimization framework g2o is employed to address this optimization challenge, which fine-tunes the camera's self-pose trajectory and the dynamic target poses as described in earlier sections, culminating in a globally consistent 3D map that integrates both dynamic and static features.

The factor graph structure of this system at discrete moments $[k-2, k]$ is depicted in FIGURE 11. This graph includes various nodes such as static feature points m^B, shown as blue rectangles, camera pose matrices X, depicted as gray rectangles, dynamic target feature points m^D, represented as orange rectangles, and dynamic target pose transformation matrices O, illustrated as green rectangles. All matrices are expressed in the global coordinate system. To optimize the camera and target poses

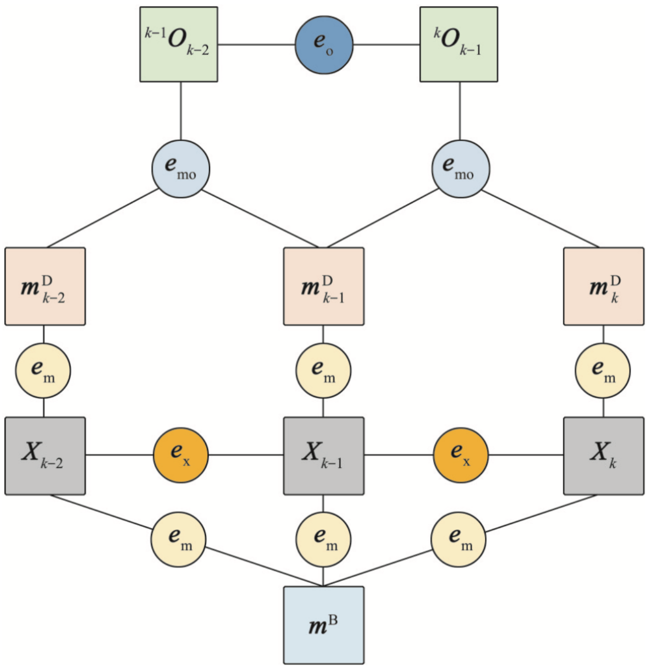


Figure 11: Graph of the optimized structure

effectively, four types of error term constraints are jointly utilized within the graph:

$$\begin{cases} e_m = X_k^{-1} \cdot m_k - \tilde{m}_k \\ e_x = (x_{k-1}^{-1} \cdot X_k)^{-1} \cdot {}^{k}V_{k-1} \\ e_{mo} = m_k^D - {}^{k}O_{k-1} \cdot m_{k-1}^D \\ e_o = {}^{k-1}O_{k-2}^{-1} \cdot {}^{k}O_{k-1} \end{cases}$$

where $\tilde{m}_k$ represents the set of 3D points involved in pose calculations at time k, and ${}^{k}V_{k-1}$ is the transformation matrix reflecting the camera's pose relative to time $k-1$. The nonlinear optimization tool, g2o, is applied to solve the minimization problem, optimizing both the intelligent agent's pose and target motion.

The mapping process of the module generates a local map containing sparse reconstructions of static feature points and a global map containing information about the unmanned agent and target. Static ORB features outside the target area, extracted during pose estimation and tracking, are used to generate a sparse point cloud of the background, as shown in FIGURE 12. The blue box represents the left-eye observation position of the unmanned agent, the red points are static 3D background landmark points previously tracked by keyframes, and the black points are newly observed map points from the current frame. If these new points are involved in the pose calculation, they will be added as new landmarks to the point cloud map.

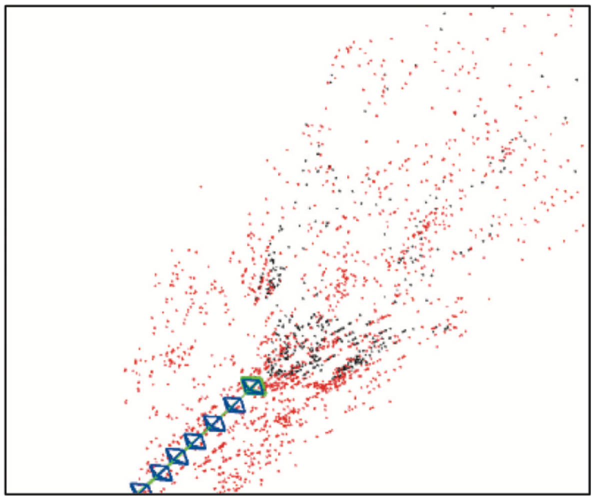


Figure 12: Map sparse reconstruction

For local map optimization, a fixed window of 15 frames is maintained. This approach conserves computational resources while ensuring that the system retains only the most relevant spatial and temporal data. The module logs the results of calculations and optimizations from all time points into a global map. As exploration progresses, this global map is continually updated, outputting refined self-pose estimates, motion trajectories of the intelligent agent, and detailed information such as types, locations, and velocities of battlefield combat units, ultimately enhancing scene perception capabilities.

5. APPLICATION TESTING IN THE BATTLEFIELD METAVERSE

The 3D environment perception system described in this paper is implemented using Python and C++ on Ubuntu 24.04 LTS. Instance segmentation and dense optical flow estimation are performed offline by the terminal using pretrained models, processing input frame sequences. The results, including segmentation data and optical flow vectors, are transmitted to the dynamic scene localization and perception module via a socket communication protocol.

To evaluate the system's feasibility and perception performance, mobile detection tests are conducted in the battlefield metaverse environment using unmanned intelligent agents. The experimental testing and evaluation in development mode compare the backend's localization and target information against ground truth, as shown in FIGURE 13. During the tests, unmanned intelligent agents navigate through dynamic scenes featuring complex background textures that blend with target camouflage. Combat units appear at high speeds, and multiple dynamic targets often emerge within the same field of view. The agents maintain high speeds, executing high-dynamic rotations within just a few frames, which poses significant challenges for traditional localization and tracking methods.

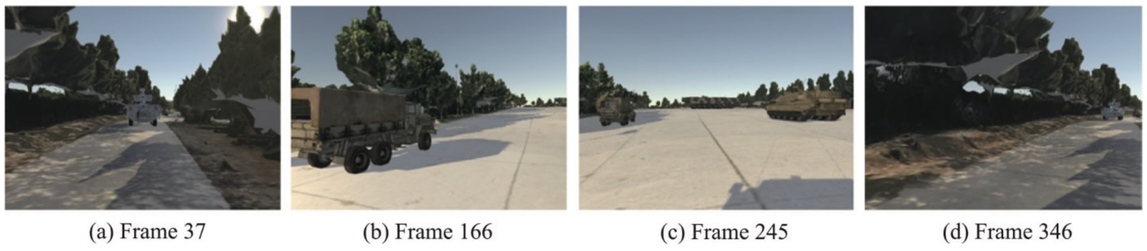


Figure 13: Frames from the experimental sequence

Prior to system operation, the binocular camera parameters mounted on the unmanned intelligent agents are determined to construct the camera intrinsic matrix K.

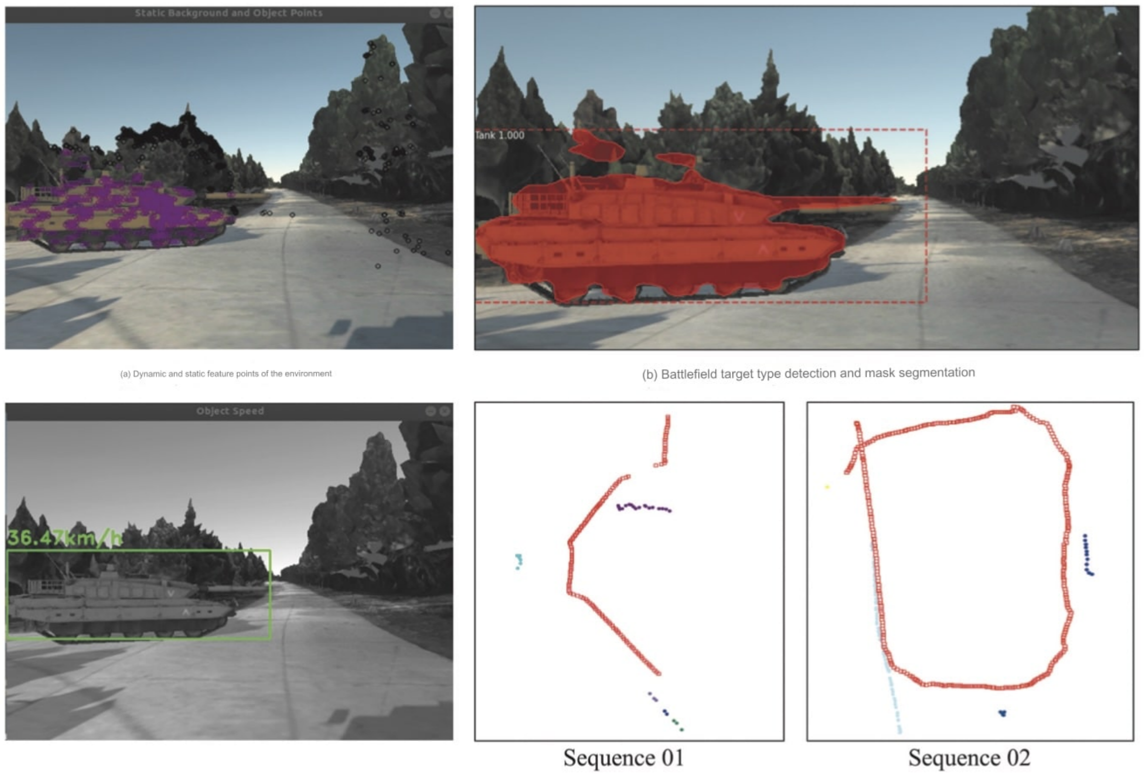

Static Background and Object Points
(a) Dynamic and static feature points of the environment
Tank 1.000
(b) Battlefield target type detection and mask segmentation
Object Speed
36.47km/h
Sequence 01
Sequence 02

Figure 14: System operation effect

Visualization of the input frame sequence for the perception system is displayed in FIGURE 14: panel (d) shows real-time trajectory mapping of camera localization and detected target positions on the xz plane, updated in a front-end window for each frame. The trajectory of the intelligent agent is marked in red, while colored dots indicate the movement trajectories of different detected targets. If no target is present in the field of view, only the intelligent agent's pose is outputted; when targets are detected, the corresponding frame outputs the type, position, speed, and other situational information of the targets.

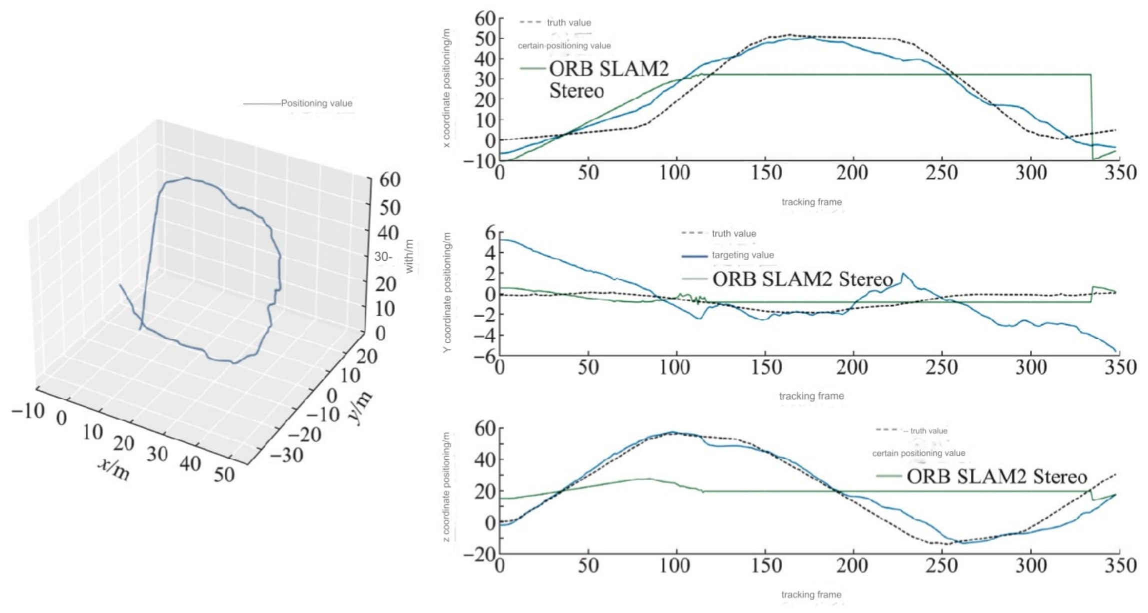
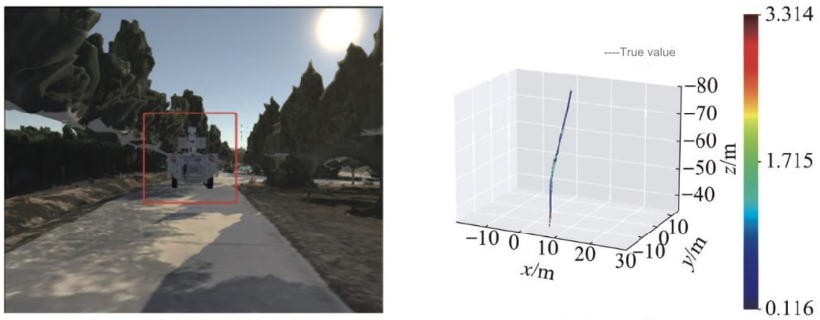


Fig.15 Comparison of intelligent body trajectory and three-axis accuracy

15 Trajectory and comparison for intelligent agent



(a) Tracking combat unit targets (b) Target global trajectory error heat map

Figure 15: Comparison of the positioning accuracy along the three axis

For the unmanned agent's own pose, the system can provide its global trajectory, as shown in FIGURE 15 (a). Additionally, FIGURE 15 (b) shows a comparison of the positioning accuracy along the xyz axes. The trajectory estimated by the system in this paper is closer to the ground truth, with a Root Mean Square Error (RMSE) of the Absolute Trajectory Error (ATE) of 3.965 meters. In comparison, the error provided by ORB-SLAM2 Stereo is 7.104 meters, representing a 44.2% improvement in accuracy. By incorporating scene priors and IMU information, the system presented in this paper can effectively solve the drift in self-localization of unmanned agents in complex dynamic scenes, compared to purely vision-based methods that assume environmental rigidity. Furthermore, the system can effectively cope with and continue localization even when ORB-SLAM2 Stereo loses tracking due to rapid rotations.

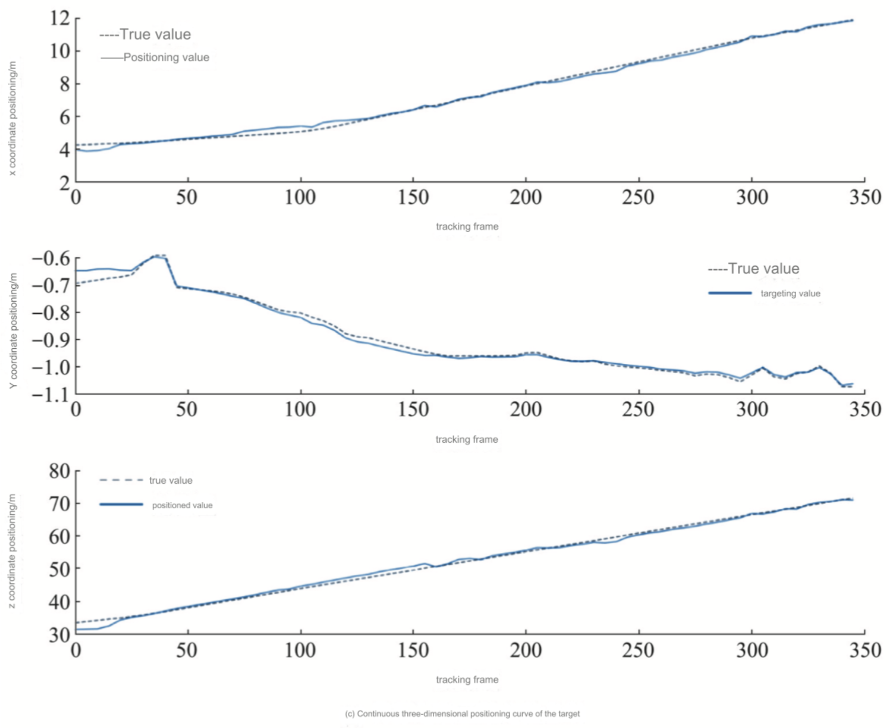


Figure 16: Comparison of target trajectory and accuracy

In terms of target perception, the system can provide the global position and trajectory of the target based on the center of the rigid body. FIGURE 16 shows the 3D trajectory of an armored transport vehicle detected in an application as an example. In the heatmap, the redder the color, the greater the drift at that position. The solid line represents the target's positioning along the three axes over time, while the dashed line represents the ground truth. From the comparison, it can be seen that over a tracking length of 85 meters, the RMSE (Root Mean Square Error) of the global positioning trajectory is approximately 1 meter, and the three-axis positions remain close to the ground truth throughout.

The output results of target perception provide three-dimensional localization, with the translation component t of the pose matrix used for error calculation. These pose and trajectory localization errors for the unmanned intelligent agent and each dynamic target in the battlefield test sequence remain low, demonstrating highly accurate estimates.

Ablation experiments compare the purely visual motion model with the estimation method that integrates IMU and optical flow. The latter significantly improves the number of tracked feature points, effectively handling rapid rotations and complex dynamic processes, thus ensuring tracking and localization accuracy. The system can separately save static feature point information, the intelligent agent's self-pose trajectory, and estimates of each target's type, position, and speed in each frame. These pieces of 3D scene perception information are outputted frame by frame, supporting situational fusion analysis and decision-making in the battlefield metaverse.

6. CONCLUSION

This paper introduces a system for dynamic 3D scene perception on battlefields, utilizing unmanned intelligent agents equipped with binocular vision and IMU devices within the battlefield metaverse. After processing binocular video streams and IMU data, deep learning-based instance segmentation and dense optical flow prediction networks were deployed. A target dataset was developed using four data augmentation methods, and the ResNet101+FPN feature extraction backbone was selected for model training. The system achieved a combat unit type recognition accuracy of 91.8%, a mean Intersection over Union (mIoU) of 0.808, and a mean Average Precision (mAP) of 0.6064, effectively outputting the type, detection box, and segmentation mask for each combat unit. Additionally, the PWC-NET was utilized to derive inter-frame dense optical flow estimates, serving as crucial priors for the dynamic scene localization and perception module. This module successfully mitigates localization interference in complex scenes, refines pose estimation using optical flow, and optimizes a global map through factor graph techniques, which continuously updates with the pose trajectory of the unmanned intelligent agent and the localization and motion information of various battlefield targets.

Application testing within the battlefield metaverse showed a 44.2% improvement in self-localization accuracy of the unmanned intelligent agent over ORB-SLAM2 Stereo. The system accurately estimated the type, localization, and motion speed of each battlefield target within the field of view. It continuously annotated dynamic and static information on the battlefield, updating the real-time trajectory of camera localization and detected target positions, and consistently fed this information back to other modules of the battlefield metaverse.

The extraction of targets by the deep learning module provides crucial situational information. Moreover, the integration of target masks and inter-frame motion allows for classification of dynamic and static feature points during the agent's self-localization process, addressing errors and drift caused by the rigidity assumption in SLAM environments, thus enhancing localization accuracy. The proposed system effectively tackles the challenges of dynamic complexity and information loss in battlefield situational awareness, offering high accuracy and comprehensive tracking capabilities. This establishes a solid foundation for subsequent battlefield situational assessments and tactical decision-making.

Future research will aim to improve networks and environment reconstruction methods to precisely and rapidly identify different models of the same type of combat units and typical buildings. By leveraging current environmental data, multi-agent collaboration could be utilized for situational generation, and adaptive planning of reconnaissance paths for unmanned intelligent agents might be implemented. Extending the perception of 3D battlefield scenes further can enhance the battlefield metaverse, improving real-time situational feedback, optimizing deep human-machine interactions, facilitating modern joint combat command and control, and guiding real-world applications.

References

[1] Lei Y, Wang Z, Chen F, Wang G, Wang P, et al. Recent Advances in Multi-Modal 3D Scene Understanding: A Comprehensive Survey and Evaluation. 2023. ArXiv preprint: https://arxiv.org/pdf/2310.15676

[2] Gothoskar N, Cusumano-Towner M, Zinberg B, Ghavamizadeh M, Pollok F, et al. 3DP3: 3D Scene Perception via Probabilistic Programming. Advances in Neural Information Processing Systems. 2021;34:9600-9612.

[3] Cheng S, Sun C, Zhang S, Zhang D. SG-SLAM: A Real-Time RGB-D Visual SLAM Toward Dynamic Scenes With Semantic and Geometric Information. IEEE Transactions on Instrumentation and Measurement. 2022;72:1-12.

[4] Liu G, Zeng W, Feng B, Xu F. DMS-SLAM: A General Visual SLAM System for Dynamic Scenes With Multiple Sensors. Sensors. 2019;19:3714.

[5] Cheng J, Wang Z, Zhou H, Li L, Yao J. DM-SLAM: A Feature-Based SLAM System for Rigid Dynamic Scenes. ISPRS International Journal of Geo-Information. 2020;9:202.

[6] Chen L, Ling Z, Gao Y, Sun R, Jin S. A Real-Time Semantic Visual Slam for Dynamic Environment Based on Deep Learning and Dynamic Probabilistic Propagation. Complex & Intelligent Systems. 2023 Oct;9(5):5653-5677.

[7] Bescos B, Fácil JM, Civera J, Neira J. DynaSLAM: Tracking, Mapping, and Inpainting in Dynamic Scenes. IEEE Robotics and Automation Letters. 2018;3:4076-4083.

[8] Rosinol A, Violette A, Abate M, Hughes N, Chang Y, et al. Kimera: From SLAM to Spatial Perception With 3D Dynamic Scene Graphs. The International Journal of Robotics Research. 2021;40:1510-1546.

[9] Beghdadi A, Mallem M. A Comprehensive Overview of Dynamic Visual Slam and Deep Learning: Concepts, Methods and Challenges. Machine Vision and Applications. 2022;33:54.

[10] Mur-Artal R, Tardós JD. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. IEEE transactions on robotics. 2017;33:1255-1262.

[11] Fleming RW. Visual Perception of Materials and Their Properties. Vision research. 2014;94:62-75.

[12] Fleming RW, Dror RO, Adelson EH. Real-World Illumination and the Perception of Surface Reflectance Properties. Journal of vision. 2003;3(5):347-368.

[13] Ilg E, Mayer N, Saikia T, Keuper M, Dosovitskiy A, et al. Flownet 2.0: Evolution of Optical Flow Estimation With Deep Networks. In Proceedings of the IEEE conference on computer vision and pattern recognition 2017:2462-2470.

[14] Ilg E, Saikia T, Keuper M, Brox T. Occlusions, Motion and Depth Boundaries With a Generic Network for Disparity, Optical Flow or Scene Flow Estimation. In Proceedings of the European conference on computer vision (ECCV). 2018:614-630.

[15] Mayer N, Ilg E, Hausser P, Fischer P, Cremers D, et al. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2016:4040-4048.

[16] Geiger A, Lenz P, Stiller C, Urtasun R. Vision Meets Robotics: The KITTI Dataset. The International Journal of Robotics Research. 2013 Sep;32(11):1231-1237.