Bridging Theory and Practice in Military Education through Advanced Technologies

Ondřej Pekař[1], Jiří Novák[1], Jaroslav Varecha[2], Jiří Šotnar[1] and Jan Drábek[1]
[1]University of Defence, Brno, Czech Republic
[2]Armed Forces Academy, Liptovský Mikuláš, Slovakia

ondrej.pekar@unob.cz
jiri.novak12@unob.cz
jaroslav.varecha@aos.sk
jiri.sotnar@unob.cz
jan.drabek2@unob.cz

Abstract: This study presents a comprehensive examination of the developmental trajectory and operational advancement of a pioneering military simulation apparatus explicitly engineered to facilitate the seamless integration of joint fire support principles within the ambit of military education, with a specific focus on artillery students. The Virtual Reality (VR) and Augmented Reality (AR) technology provided by industry leaders such as Microsoft and Meta present solutions for interdisciplinary pursuits situated at the confluence of emergent technologies and contemporary military instructional methodologies. Central to the efficacy of this endeavour is the incorporation of meticulously crafted virtual and augmented reality models of authentic military equipment to imbue the simulation experience with unparalleled realism and fidelity to the principles of artillery fires. Collaborative ventures with private enterprises have facilitated the production of a simulator characterized by its gamification elements and unparalleled verisimilitude, enhancing its efficacy as an educational tool. Augmented by spatial intelligence gleaned from Unmanned Aerial Vehicle (UAV) sensors, the simulation framework can be further enriched by integrating true-to-life topographic representations meticulously synthesized by amalgamating laser scanning methodologies. Positioned as an indispensable cornerstone within the academic purview of the esteemed Department of Fire Support at the University of Defence, this initiative aspires to impart foundational knowledge and cultivate a proactive ethos of inquiry and engagement amongst the students. Furthermore, the simulator is a pivotal instrument for military training, offering a risk-free environment devoid of the hazards associated with live artillery fire exercises. By mitigating financial burdens and other logistical constraints inherent in conventional training methodologies, this innovative simulation apparatus represents a paradigm shift in contemporary military education and training practices.

Keywords: Simulation, Artillery, Training, VR, AR

1. Introduction

Artillery live fire training is indispensable for fire-observing units due to its unparalleled benefits (Blaha, Potužák, Šustr, *et al.* 2021). It provides soldiers with a realistic environment, enabling them to experience and respond to combat scenarios and improving their decision-making skills. Providing joint fire support to maneuvering forces involves coordinating diverse elements such as artillery, close air support, and naval forces, all operating under stringent principles of precision and synchronization, while assessing the effects on the target, see Fig. 1. However, live fire training also presents challenges such as safety risks, logistical complexities, and environmental impacts, which can be mitigated through simulation technologies (Foltin, Vlkovský, Mazal, *et al.* 2018).

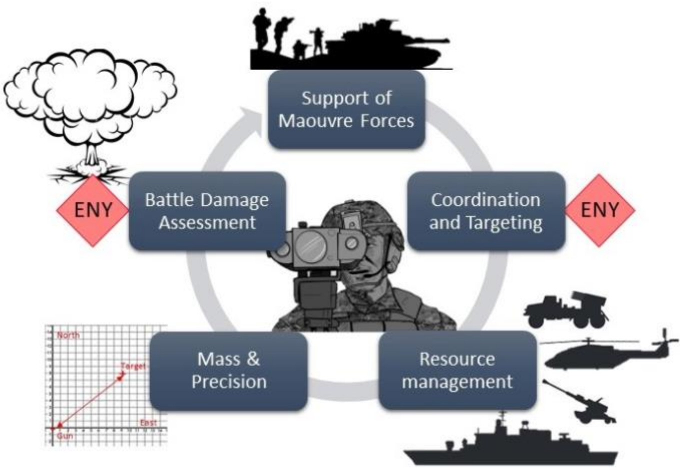

Figure 1 Complexity and principles of Joint Fire Support

However, simulation-based training may lack the realism and sensory immersion crucial for accurately replicating real-world combat situations, potentially impacting soldier's readiness (Mazal, Zezula, Procházka, *et al.* 2022). Acknowledging these limitations, collaborations between institutions like the University of Defence and the YORD company leverage Virtual Reality (VR) and Augmented Reality (AR) technologies. These efforts aim to develop immersive training environments that bridge theoretical education with practical applications, enhancing training effectiveness and engagement.

This paper introduces the architecture of the simulator software suite and discusses key developmental challenges encountered.

2. Architecture

The simulator suite comprises two main applications (Control and Field) and two additional modules for supporting functionality. This design simplifies application architecture, enables independent updates, reduces code complexity, and facilitates the reuse of common functions.

The control module is a standalone application designed for personal computers running Microsoft Windows. It allows access to local files via the system-native file picker dialogue. The application's main functions are to load scenarios from the backend, modify area and target definitions, and display field users' real-time locations. Users can save scenarios locally for future use. The control application utilizes *Mapbox's Unity plugin* to provide three-dimensional map data with multiple terrain layers (Mapbox 2024).

Unity's glTFast plugin loads 3D models in gltf/glb format at runtime to display three-dimensional targets and area ground models, reducing application size and enabling independent model distribution (Unity Technologies 2024e). Draco compression via the *DracoUnity plugin* minimizes network bandwidth, which is crucial for areas with limited mobile network throughput where the Field application may operate (Atteneder 2019). *Unity's Universal Rendering Pipeline* (URP) supports scalable graphical tools and PBR materials from glTF files, enhancing visual quality and realism in the simulation (Unity Technologies 2024f).

The field module allows users on compatible mobile phones and VR headsets to view targets in a designated area. Users navigate the area using the mobile phone version and can then switch to augmented reality (AR) mode on either the mobile phone or a connected VR headset. In addition to various plugins and libraries mentioned earlier, we utilize *Apple's ARKit plugin* (via AR Foundation) and Meta's Core SDK plugins for positional tracking on iPhones and Quest headsets, respectively (Unity Technologies 2024a).

The simulator's backend module facilitates communication between multiple instances of field and control applications running simultaneously. It leverages *Express.js*, a *Node.js* backend framework, for this purpose and for storing target and ground model files (StrongLoop, IBM 2017; OpenJS Foundation, 2024). The backend also provides API endpoints for file listing and timestamps, manages structures for model destroy rules, and pairs mobile phones to headsets.

The networking module contains definitions of structures shared between the field and control applications. It also allows for two-way communication by utilizing APIs exposed by the backend. Communication is facilitated using *Socket.IO*, which allows for real-time messaging between connected peers (itisnajim 2022).

In addition to APIs provided by our backend modules, *Mapbox's Raster Tiles API* is used to obtain elevation information for accurately placing targets and areas. We discuss the reasoning behind this decision in a further section. The workflow consists of downloading a specific tile texture and analyzing the colour value of a specific point (MapBox 2024). The RGB channels are then decoded to obtain the accurate altitude. In order to save on network bandwidth, we have created a file synchronization system. Since 3D models used within the applications can potentially reach hundreds of megabytes in size, we download them once and then store them in the local storage on the device. Each time the model is requested, we ask the server to provide us with a timestamp of the remote file, and if said timestamp is older than the timestamp of the local file, we skip redownloading the model. Created models of enemy air units and models of terrain are shown in Figure 1 (Cimr, Tomaskova, Cimler, *et al.* 2018).

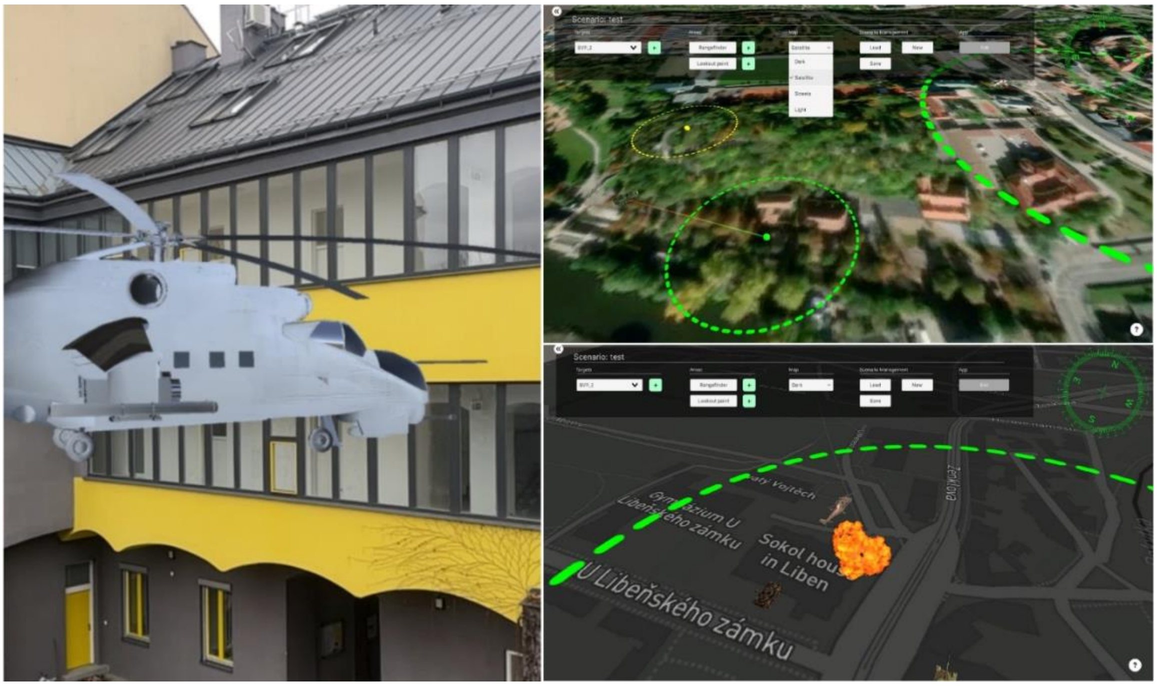

Figure 2: Display of models created within the terrain [authors]

3. Encountered Issues

During the development of the simulation, several challenges were encountered and addressed. These included issues related to GPS and vertical inaccuracy, the absence of a compass on the Quest 3 headset, communication and pairing between devices, implementation of the rangefinder functionality, and UAV terrain modelling. Strategies such as leveraging alternative orientation methods, optimizing communication protocols, and refining data acquisition techniques were employed to mitigate these challenges effectively, ensuring the accuracy and reliability of the simulation system. These experiences highlight the importance of adaptive problem-solving and innovative approaches in overcoming technical hurdles during simulation development.

3.1 GPS and Vertical Inaccuracy

Utilizing GPS-enabled mobile phones for user location presents challenges due to inherent inaccuracies and noise in location data, especially in varying environments. While employing specialized GPS receivers could improve accuracy, it adds complexity to the system architecture. Inaccurate GPS measurements affect user experiences in control and field applications, leading to position "jumping" and drift effects in rendered models.

To address these issues, we adopted a strategy of taking location measurements only upon entering AR mode, reducing drift effects. Additionally, we leverage Unity's Location API's horizontal accuracy property for enhanced location precision in real-time reporting to control users (Unity Technologies 2024c).

Elevation data accuracy posed a significant challenge, with errors exceeding 10 meters using Unity's altitude property—an interim solution using *Mapbox's UnityTile.QueryHeightData* method provided relative elevation but lacked absolute accuracy. We improved accuracy by leveraging Mapbox's Raster Tile API, which provides exact elevation data encoded in RGB channels, offering accurate measurements down to centimetres (MapBox 2024).

3.2 Missing Compass on Quest 3

Initially, we expected GPS data to be provided by the mobile device since the Quest 3 lacks a GPS module. We assumed the device would have a compass module accessible via Unity's Compass API. However, despite numerous modifications to the AndroidManifest, we needed help getting the compass working on the Quest 3. As a result, we devised a solution where the user runs the application simultaneously on both the headset and the mobile phone. They orient the devices to face the same direction and then align themselves with the north direction. Once in AR mode, users can stow away their mobile phones, continuing to use the application on the headset. Users must recalibrate the orientation if the headset goes into sleep mode (Unity Technologies 2024b).

3.3 Device Communication and Pairing

Initially, our system relied on local network connections between mobile phones and headsets to minimize network latency during critical operations like heading updates in calibration. The mobile phone acted as an intermediary between the backend and headsets, especially for headsets lacking direct mobile data access. We utilized *Mirror* for local networking but faced challenges with peer discovery when headsets were connected via mobile hotspots, requiring additional hardware like an external network access point. We shifted all data communication to the backend to address these issues, albeit introducing slight delays in heading updates during calibration (UNET 2024).

We sought automatic linking between individual mobile phones and headsets to seamlessly support multiple field users without requiring a headset-side input interface. For this purpose, we leveraged Unity's SystemInfo API, specifically the *deviceUniqueIdentifier* property. However, we encountered issues as this identifier can change under specific conditions, such as app updates on both iOS and Android. To mitigate this, we store the initial identifier value and reference it for consistency (Unity Technologies 2024d).

3.4 Rangefinder Implementation

To replicate the real hardware experience, we designed the rangefinder functionality in our system to closely resemble actual rangefinder operations. Using Unity's RenderTexture feature, we create a monoscopic view from a Camera, simulating the rangefinder's view in front of the user. The zoom level adjustment mimics natural optical settings by changing the camera's focal length, enhancing realism. For ease of zoom control on headsets, we utilize the volume buttons as accessible physical controls, although this triggers the system UI for volume adjustment, which cannot be suppressed.

Upon loading the rangefinder view, users encounter a ground model (if available), although discrepancies between the model and actual terrain may exist. To mitigate this, we automatically adjust target heights to align with the ground model's geometry, assuming all targets are positioned on the ground level for consistency (Unity Technologies 2018).

4. Terrain Modeling

In our endeavour to create a terrain model for AR/VR simulation using terrain data acquired by the DJI M300 RTK UAV and Zenmuse L1 LiDAR (Light Detection and Ranging) sensor, we encountered challenges primarily related to data processing and the realism of tree representations. Integrating data from laser scanning provided detailed 3D point cloud true-coloured data for constructing an accurate terrain model. However, issues arose with representing trees realistically in the simulation, as the initial models appeared unnatural and lacked lifelike features, as shown in Figure 2. Addressing these challenges involved refining data processing algorithms and enhancing tree modelling techniques to improve visual fidelity and create a more immersive virtual environment for AR/VR simulations (Ivan, Sustr, Pekar, *et al.* 2022).

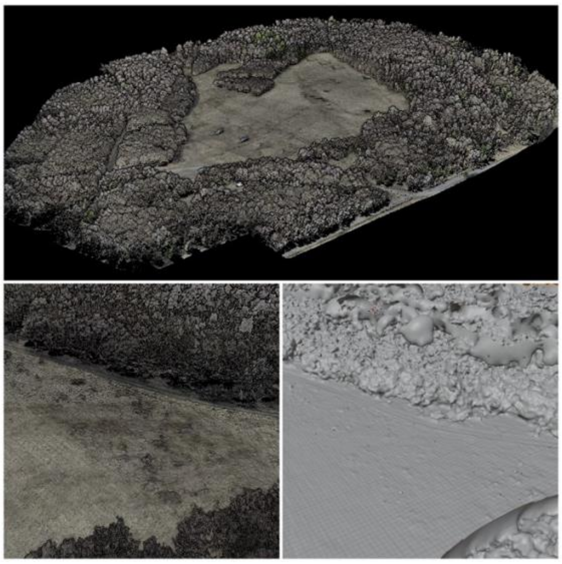

Figure 3: Acquired point cloud data and model terrain derived [authors]

5. Conclusion

The development of the simulator suite marks a significant advancement in military education and training methodologies, particularly in the context of fire support principles for artillery students. Our comprehensive approach, integrating Virtual and Augmented Reality (VR/AR) and prospects of unmanned aerial vehicle (UAV) sensors, results in a robust simulation framework capable of delivering unparalleled realism and fidelity to artillery fire principles. Despite the challenges, these experiences underscore the importance of leveraging emergent technologies and interdisciplinary collaboration to overcome technical hurdles and advance military education and training practices. The ongoing refinement of data processing algorithms and tree modelling techniques reflects our commitment to continuous improvement and delivering an immersive and compelling training environment for military personnel.

References

Atteneder, A. (2019) *DracoUnity*. [Online]. 2019. GitHub. Available at: https://github.com/atteneder/DracoUnity (Accessed: 9 April 2024).

Blaha, M., Potužák, L., Šustr, M., Ivan, J., et al. (2021) Simplification Options for More Efficient Using of Angular and Linear Measuring Rules for Fire Control. *International Journal of Education and Information Technologies*. [Online] 1528–34. Available at: doi:10.46300/9109.2021.15.4.

Cimr, D., Tomaskova, H., Cimler, R., Kuhnova, J., et al. (2018) A System to Evaluate an Air-Strike Threat Level Using Fuzzy Methods. In: *Computational Collective Intelligence*. [Online]. Cham, Springer International Publishing. pp. 322–331. Available at: doi:10.1007/978-3-319-98446-9_30 (Accessed: 9 April 2024).

Foltin, P., Vlkovský, M., Mazal, J., Husák, J., et al. (2018) Discrete Event Simulation in Future Military Logistics Applications and Aspects. In: *Modelling and Simulation for Autonomous Systems*. [Online]. Cham, Springer International Publishing. pp. 410–421. Available at: doi:10.1007/978-3-319-76072-8_30 (Accessed: 9 April 2024).

itisnajim (2022) *SocketIOUnity*. [Online]. 2022. GitHub. Available at: https://github.com/itisnajim/SocketIOUnity (Accessed: 9 April 2024).

Ivan, J., Sustr, M., Pekar, O. and Potuzak, L. (2022) Prospects for the Use of Unmanned Ground Vehicles in Artillery Survey. In: *Proceedings of the 19th International Conference on Informatics in Control, Automation and Robotics*. [Online]. 14 July 2022 SCITEPRESS - Science and Technology Publications. pp. 467–475. Available at: doi:10.5220/0011300100003271 (Accessed: 16 August 2023).

MapBox (2024) *Access elevation data*. [Online]. 2024. Mapbox. Available at: https://docs.mapbox.com/data/tilesets/guides/access-elevation-data/ (Accessed: 9 April 2024).

Mapbox (2024) *Maps for Unity*. [Online]. 2024. Mapbox. Available at: https://www.mapbox.com/unity (Accessed: 9 April 2024).

Mazal, J., Zezula, J., Procházka, J. and Procházka, D. (2022) The use of modeling and simulation in the development process optimization of the Armed Forces of the Czech Republic. *Vojenské rozhledy*. [Online] 31 (4), 140–158. Available at: doi:10.3849/2336-2995.31.2022.04.140-158.

OpenJS Foundation, (2024) *Nodejs*. [Online]. 2024. Nodejs. Available at: https://nodejs.org/en/ (Accessed: 9 April 2024).

StrongLoop, IBM (2017) *Express*. [Online]. 2017. Express. Available at: https://expressjs.com/ (Accessed: 9 April 2024).

UNET (2024) *Mirror*. [Online]. 2024. Mirror-networking. Available at: https://mirror-networking.com/ (Accessed: 9 April 2024).

Unity Technologies (2024a) *Apple ARKit XR Plug-in*. [Online]. 2024. Docs.unity3d. Available at: https://docs.unity3d.com/Packages/com.unity.xr.arkit@5.0/manual/ (Accessed: 9 April 2024).

Unity Technologies (2024b) *Compass.trueHeading*. [Online]. 2024. Docs.unity3d. Available at: https://docs.unity3d.com/ScriptReference/Compass-trueHeading.html (Accessed: 9 April 2024).

Unity Technologies (2024c) *LocationInfo*. [Online]. 2024. Docs.unity3d. Available at: https://docs.unity3d.com/ScriptReference/LocationInfo.html (Accessed: 9 April 2024).

Unity Technologies (2024d) *SystemInfo.deviceUniqueIdentifier*. [Online]. 2024. Docs.unity3d. Available at: https://docs.unity3d.com/ScriptReference/SystemInfo-deviceUniqueIdentifier.html (Accessed: 9 April 2024).

Unity Technologies (2024e) *Unity glTFast*. [Online]. 2024. Docs.unity3d. Available at: https://docs.unity3d.com/Packages/com.unity.cloud.gltfast@6.2/manual/index.html (Accessed: 9 April 2024).

Unity Technologies (2024f) *Universal Render Pipeline*. [Online]. 2024. Docs.unity3d. Available at: https://docs.unity3d.com/Packages/com.unity.render-pipelines.universal@17.0/manual/index.html (Accessed: 9 April 2024).

Unity Technologies (2018) *Using Physical Cameras*. [Online]. 2018. Docs.unity3d. Available at: https://docs.unity3d.com/Manual/PhysicalCameras.html (Accessed: 9 April 2024).