Improved Robotic Platform to perform Maintenance and Upgrading Roadworks: The HERON Approach

Grant Agreement Number: 955356

D5.4: Development of the AR components

Work package	WP5: Development of the Robotic Platform, Improved Navigation and integration with the sensing devices
Activity	Task 5.5: Development of the AR components, UI development and integration
Deliverable	D5.4: Development of the AR components
Authors	Dimitris Papageorgiou, Giannis Apostolatos, Christos Polykretis, Aggelos Aggelis
Status	Final (F)
Version	1.0
Dissemination Level	Public (PU)
Document date	30/06/2024
Delivery due date	31/03/2024
Actual delivery date	30/06/2024
Internal Reviewers	Nikos Frangakis (IKH), Elena Avatangelou (INAC)
	This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under grant agreement no 955356.

Document Control Sheet

Version history table			
Version	**Date**	**Modification reason**	**Modifier**
0.1	25/04/2024	Initial Table of Contents and basic structure of the deliverable	Christos Polykretis
0.2	13/05/2024	Initial draft	Christos Polykretis, Aggelos Aggelis
0.3	21/05/2024	Components overview and system dependencies sections	Dimitris Papageorgiou
0.4	22/05/2024	UI and implementation sections	Giannis Apostolatos
0.5	27/05/2024	Updated deliverable content	Christos Polykretis
0.6	29/05/2024	Updated deliverable content	Aggelos Aggelis
0.7	31/05/2024	Initial deliverable version ready for internal review	Christos Polykretis
0.8	21/06/2024	Internal review comments	Nikos Frangakis
0.9	26/06/2024	Applied review comments	Christos Polykretis, Aggelos Aggelis
1.0	30/06/2024	Final version ready for submission	Christos Polykretis

Legal Disclaimer

Table of Contents

List of Tables

List of Figures

Abbreviation List

Abbreviation	Definition
AI	Artificial Intelligence
API	Application Programming Interface
AR	Augmented Reality
ASGI	Asynchronous Server Gateway Interface
CPU	Central Processing Unit
COP	Common Operational Picture
DF	Data Fusion
FOV	Field Of View
GPS	Global Positioning System
HLP	High Level Planner
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
IMS	Incident Management System
IMU	Inertial Measurement Unit
IoT	Internet of Things
IT	Information Technology
MR	Mixed Reality
OPM	Object Relational Mapping
PC	Personal Computer
REST	Representational State Transfer
RI	Road Infrastructure
SDK	Software Development Kit
UI	User Interface
UGV	Unmanned Ground Vehicle
UX	User Experience
VR	Virtual Reality
XR	Extended reality

Executive Summary

AR is a transformative technology that overlays digital elements (e.g., computer-generated images and sounds) and other sensory enhancements onto the real world, advancing its users' perception and interaction with their surrounding environment. After significant progress achievements over the past decade on how we interact with digital content, today, AR is utilized in various applications and sectors, including gaming, education, healthcare, and retail. Despite their vast potential, AR faces several challenges, including technological limitations, high costs, and privacy concerns. However, ongoing advancements in its hardware and software components are gradually addressing these issues, making AR technology more accessible and practical for use.

AR glasses are wearable devices that are typically worn like regular glasses, and enable users to view and interact with augmented content directly in their line of sight. Unlike traditional screens, these glasses provide an immersive experience by overlaying the digital elements onto the physical environment, allowing for hands-free access to information and applications. One of the key advantages of AR glasses is their potential to enhance productivity and efficiency in various professional settings.

On the basis of AR capabilities, a set of AR glasses was obtained and exploited, and a relevant app's UI (with related applications for real-time information visualization) was developed under the HERON project, in order to enhance the work efficiency and improve the road maintenance operations implemented by the RI supervisors and personnel in the field.

This document constitutes the fourth deliverable of WP5, namely "Development of the AR components", of the HERON project under Grant Agreement No. 955356. The specific deliverable D5.4 refers to the outcome of Task 5.5 involving the "Development of the AR components, UI development and integration", and covering the HERON project's period of M20-M34.

1 Introduction

1.1 Purpose of the Document

Due to its particular type defined as “OTHER”, the purpose of this document is to provide a description of the AR technical components, including both hardware (AR glasses) and software (app’s UI), which constitute the actual deliverable (outcome) of the Task 5.5. Considering the users’ needs and requirements in terms of HERON project,their interactive implementation capabilitiesare also presented.

The specific document is organized in totally eight (8) sections. Except for the current section 1, section 2 refers to the HERON high-level architecture, focusing on the AR components. Section 3 provides an overview of the AR components and their development framework. Section 4 presents in more detail the UI for the AR glasses and its respective capabilities. Section 5 allows the users to recognize the dependencies of the developed AR components. Section 6 describes the AR component implementation capabilities for on-field road maintenance operations. Section 7 provides an overview of the contribution of AR components as a result of their integration to overall HERON system. Finally, section 8 includes the conclusions.

1.2 Intended Audience

This specific deliverable report is public and thus can be accessed by any interested stakeholder. Among others, this is of exceptional interest to the end-users, namely the road agencies responsible for the RI operation and management. Finally, it is worth noting that other envisioned stakeholders interested in using and/or developing AR technological tools can be also considered within the document’s target group.

1.3 Interrelations

First of all, this deliverable interacts with the WP2 entitled “End-Users Requirements, Metrics and System Design”; the outcomes of its deliverables D2.1, D2.2 and D2.3, namely “User Requirements, definition of the Use Cases and KPIs”, “Design of the overall system architecture” and “Geographic Data and Services inventorying and Open Data repositories gathering”, respectively, provide specific guidelines and requirements which were taken into account during the development of the AR components. In particular, the D2.1 refers to the end-users’ requirements on the basis of analyzing current practices, needs, and expectations from the involved road agencies. The D2.2 determines the overall HERON system architecture including the specifications of the different technical components developed for HERON project. Moreover, the D2.3 reports the specifications of various road-related data that was available by the HERON involved road agencies. Hence, it can be noted that these three deliverables involve the challenges, requirements and limitations that the AR components should face, satisfy and overcome, respectively, in terms of HERON system integration.

In addition, the autonomous robot with on-board and drone sensors’ perceptual abilities, developed in the previous four (4) Tasks of WP5 (T5.1-5.4), is intended to be further reinforced with the AR components’ perceptual and operational abilities. Moreover, under the integrated framework, the AR components will be integrated with the HERON IMS/COP platform (WP6) in order to assure not only a coordinated response and decision making, but also to visualize the amount of the existing defects leading to minimum risk, low cost and highly effective road intervention. The developed AR components as part of the HERON integrated system is going to be also rigorously tested and evaluated in the field during the pilot site demonstrations (WP7).

2 AR Components in the HERON System

HERON project aims to develop an integrated automated system to perform road maintenance and upgrading operations, such as sealing cracks, patching potholes, painting markings, and autonomous replacement of CUD elements, but also supporting the pre- and post-intervention phase (including visual inspections, as well as dispensing and removing traffic cones in an automated and controlled manner).
The HERON system consists of various sub-systems which interact each other [1]. These sub-systems are the following:

i. **autonomous ground robotic vehicle** that will be supported by autonomous **drones** to coordinate maintenance operations and the pre/post- intervention phase;
ii. **various robotic equipment**, including sensors and actuators (e.g. tools for cut and fill, surface material placement and compaction, modular components installation) placed on the main vehicle;
iii. **sensing interface** installed both on the robotic platform and to the RI to allow improved monitoring of the structural and functional conditions;
iv. **control software** that interconnects the sensing interface with the actuating robotic equipment;
v. **AR visualization tools** that enable the maintenance crew to see in detail surface defects and markings under survey;
vi. **AI-based toolkits** that will act as the middleware of a twofold role for:
    a. optimally coordinating the road maintenance/upgrading workflows, and
    b. intelligent processing of distributed data coming from the vehicle and the infrastructure sensors for safe operations and not disruption of other routine operations or traffic flows;
vii. **enhanced visualisation user interface** which supports decisions;
viii. **communication modules** to allow for Vehicle-to-Infrastructure/Everything data exchange for predictive maintenance and increase users' safety.

As it is mentioned above, **AR visualization tools** will be integrated into the HERON system. These tools, which this deliverable is related to, consist of AR glasses and their respective UI for the robot operators and RI maintenance supervisors. They have been designed to host and provide related applications (apps) that overlay real-time visual information on the surrounding environment (e.g., road, pavements and other areas), in order to display possible hidden structural and/or functional elements as well as additional damages.

The key benefits of AR components to the HERON system include:

- Providing a detailed and enriched view of the maintenance site, allowing for better decision-making and oversight (enhanced operational awareness).
- Visualization of real-time data and information that increase the efficiency and precision of road maintenance operations.
- Remote monitoring through AR reduces the need for physical presence on the site of maintenance operations, enhancing safety for RI personnel (supervisors and workers).

3 Overview of AR Components

The AR Components, such as the AR glasses (Magic Leap 2), the relative software (Unity 3D Engine & External Components), as well as their (interrelated) development methodology will be described in this section.

3.1 Hardware and Software Tools

3.1.1 Magic Leap 2 glasses

The Magic Leap 2 [2] glasses mark a significant advancement in the realm of AR, offering a refined and sophisticated experience tailored for enterprise applications. They are a state-of-the-art model for pass-through augmented reality applications. Enhanced ergonomic design and engineering makes these glasses suitable for extended daily use. The increased capabilities of the glasses both in indoor and outdoor settings make them a powerful tool across environments and conditions. **Magic Leap Dynamic Dimming** technology makes AR content more legible and solid across a variety of lighting conditions, such as direct sunlight, as is the case in HERON project.

Figure 1: The Magic Leap 2 AR glasses and equipment [2].

Under the hood, the Magic Leap 2 is powered by a more robust processor and increased memory capacity, enabling it to handle complex AR applications with greater efficiency. The available hardware equipment and sensor variety of Magic Leap 2 glasses contribute to an amazing degree of precision and persistence of digital content around physical objects. This model includes:

- World Sensing Cameras
- Ambient Light Sensors
- Depth Sensors
- IMU Sensors
- Eye Tracking Cameras
- Microphones
- Altimeters
- Other types of sensors

The processing of sensor data takes place in a dedicated processing unit with state-of-the-art capabilities in graphics processing. Numerous software features make use of the aforementioned sensors, including Spatial Mapping, Voice Commands, Hand Tracking & Gestures, Iris ID, Eye Tracking, Perception, Marker Tracking, Spatial Anchors, Depth maps, and many more.

The upgraded tracking sensors and cameras enhance the precision of AR interactions, offering more accurate environmental mapping and object recognition. This makes Magic Leap 2 glasses particularly effective for intricate tasks in fields such as healthcare, manufacturing, and logistics, where precision and reliability are paramount. Additionally, the device's improved battery life supports longer usage sessions, reducing the downtime caused by frequent recharging.

Moreover, industry-leading optics offer up to 70° diagonal FOV, which means plenty of the surrounding area (such as roads and maintenance sites) can be fully augmented inside our application. This broader view is complemented by a higher resolution and brighter display, making digital overlays appear more vivid and realistic. They also provide users with a more immersive and expansive visual experience, along side Dynamic Dimming, which allows the lenses to adjust opacity seamlessly. This facilitates smooth transitions between augmented and mixed reality environments.

Magic Leap 2 also shines in its enterprise-centric ecosystem, offering comprehensive support for business and industrial applications. The availability of robust developer tools and SDKs empowers developers to create custom AR solutions tailored to specific industry needs. Enhanced connectivity features, including support for Wi-Fi 6 and Bluetooth 5.0, ensure fast and reliable connections for data transfer and peripheral devices. This seamless integration into existing enterprise workflows makes Magic Leap 2 a valuable tool for organizations looking to leverage augmented reality to boost productivity, streamline operations, and foster innovation.

Some the most important technical features of Magic Leap 2 glasses are presented below.

Table 1: Main technical features of Magic Leap 2 glasses.

Technical Features	Description
Processor / Memory	Upgraded powerful processor and increased memory capacity for handling complex AR applications efficiently.
Field of View (FOV)	70° field of view, offering a wider and more immersive visual experience.

Dynamic Dimming Technology	Lenses that dynamically adjust opacity for smooth transitions between different reality environments.
Advanced Tracking Sensors	Enhanced tracking sensors and cameras for precise environmental mapping and object recognition.
Connectivity	Support for Wi-Fi 6 and Bluetooth 5.0, ensuring fast and reliable connections for data transfer and peripheral devices.

3.1.2 Unity 3D Engine

Unity [3] is a powerful game engine used for developing AR experiences, providing robust support for Magic Leap 2 integration. Unity's real-time 3D development engine lets designers and developers collaborate to create amazing immersive and interactive AR experiences. It is well-known for its outstanding benefits, such as reusing code and components from other projects or libraries using the prefab system or making complicated settings by building several other components. Another great advantage is the extensive collection of resources that are accessible. Even seasoned developers can benefit greatly from the community's time savings and knowledge. This is especially useful in complex scenarios where multiple components need to interact interchangeably, such as the HERON project. Its user-friendly interface, comprehensive set of tools, and extensive support community make it an ideal choice for creating a wide range of applications, including video games, simulations, and VR and AR experiences.

Key features of Unity 3D engine are the user-friendly scene view that helps users place objects in the 3D space, various toolbars and menus for creating 3D object of various types and forms, project window that shows all assets available to the application, multiple shaders, renderers and material to choose from, as well as easy animation management of the 3D objects. Finally, a powerful scripting engine has always been an important asset in customizing an application to the needs of every project.

One of the key strengths of Unity 3D is its cross-platform compatibility. The engine supports development for a wide array of platforms, including PC, Mac, consoles (such as PlayStation, Xbox, and Nintendo Switch), mobile devices (iOS and Android), and emerging technologies like VR and AR headsets. This flexibility allows developers to create a single project and deploy it across multiple platforms, significantly reducing development time and costs.

Unity 3D's powerful rendering engine is another major highlight. It supports both 2D and 3D graphics, providing developers with the tools to create visually stunning and highly optimized content. The engine includes a robust set of features, such as real-time global illumination, physically-based rendering, and customizable shaders, enabling developers to achieve high levels of realism and visual fidelity. Additionally, Unity's Asset Store offers a vast library of pre-made assets, plugins, and tools that can be easily integrated into projects, further streamlining the development process.

The engine is also known for its extensive scripting capabilities, primarily using C#. Unity's scripting API allows developers to create complex behaviors, game mechanics, and interactive elements with ease. The integration of Visual Studio IDE provides a powerful environment for writing and debugging code. Furthermore, Unity supports a range of third-party libraries and frameworks, giving developers the flexibility to incorporate additional functionalities and customize their projects to meet specific needs.

In summary, Unity 3D is a powerful and flexible game engine that has become a cornerstone of modern interactive content development. Its cross-platform capabilities, advanced rendering features, robust scripting tools, and supportive community make it an ideal choice for developers aiming to create high-quality applications for a variety of platforms. Whether for games, simulations, or VR/AR experiences, Unity 3D provides the tools and resources needed to bring creative visions to life.

Some the most important technical features of Unity 3D for AR applications are presented below.

Table 2: Main technical features of Unity 3D Engine.

Technical Features	Description
AR Foundation	A framework that allows developers to build cross-platform AR applications with a unified API, supporting ARKit (iOS) and ARCore (Android).
XR Interaction Toolkit	Provides a set of components for building interactive AR experiences, including features for gesture recognition, object manipulation, and user interface integration.
Real-Time 3D Rendering	Supports advanced 3D rendering techniques like real-time global illumination, physically-based rendering, and customizable shaders for creating realistic AR visuals.
Integrated AR Workflow	Seamless integration with AR-specific tools and services, such as Vuforia, ARKit, and ARCore, to streamline the development process and enhance functionality.
Multi-Platform Support	Enables deployment across various AR-enabled devices and platforms, including mobile devices (iOS and Android) and AR headsets (e.g. Magic Leap 2), ensuring broad accessibility and reach.

3.1.3 AR Foundation & MLSDK

AR Foundation [4] is a robust framework within Unity 3D that simplifies the development of AR applications across multiple platforms. It serves as a bridge between Unity and the underlying AR platforms, such as ARKit [4] for iOS and ARCore [4] for Android, providing a unified API that abstracts the complexities of working with different AR technologies. This abstraction layer allows developers to write code once and deploy it on both iOS and Android devices, significantly reducing development time and effort.

AR Foundation is built for AR development, allowing us to build rich experiences once, then deploy across multiple mobile and wearable AR devices. That feature enables developers to build and test an application on different devices, even different architectures, without an alteration in the performance or the source code of the app. In an AR Foundation project, we choose which AR features to enable by adding the corresponding manager components to our scene. Some features include Device tracking, Cameras, Plane detection, Image tracking, Object tracking, Face tracking, Body tracking, Point clouds, Raycasts, Anchors, Meshing, Environment probes, Occlusion and many more.

AR Foundation also provides comprehensive tools for managing AR sessions. It includes components for handling AR-specific events, such as changes in the tracked state of an object or updates to the AR environment. These components allow developers to create responsive and dynamic AR experiences that can adapt to the user's surroundings in real time. Furthermore, AR Foundation's session management tools ensure that AR applications can maintain performance and stability across different devices and environments, providing a consistent user experience.

The integration of AR Foundation with Unity's powerful development environment is another significant advantage. Developers can leverage Unity's extensive asset store, rich scripting capabilities, and real-time rendering engine to create high-quality AR applications. Unity's user-friendly interface and comprehensive documentation make it accessible for both novice and experienced developers. Additionally, AR Foundation benefits from Unity's active community and regular updates, ensuring that developers have access to the latest AR features and improvements.

When we build and run our app on an AR device, AR Foundation enables these features using the platform's native AR SDK, so we can create once and deploy to the world's leading AR platforms, including Magic Leap 2 glasses. The AR Foundation package contains interfaces for AR features, but doesn't implement any features itself. To use AR Foundation on a target platform, we also need a separate provider plug-in package for that platform. For HERON project, we choose the native **Magic Leap C SDK (MLSDK)** as provided by the manufacturer.

MLSDK is the official software development kit from Magic Leap, offering essential libraries and tools for building AR applications on Magic Leap 2. This framework is specifically tailored to the capabilities of Magic Leap 2 glasses and thoroughly tested by Magic Leap. It gives access to all components of the platform, such as Camera Capture, Controller, CV Camera, Depth Camera, Eye Tracking, Gesture Classification, Hand Tracking, Haptics, Meshing, Plane Detection, Segmented Dimmer, WebRTC, WebView and more. This enables the creation of immersive AR experiences where digital content seamlessly integrates with the real world. Additionally, the SDK includes advanced functionalities like hand tracking and gesture recognition, essential for building intuitive and natural user interfaces, where users can manipulate virtual objects with simple hand movements.

Moreover, the Magic Leap SDK integrates smoothly with popular development environments like Unity and Unreal Engine, making it accessible to a broad range of developers. This integration allows developers to use familiar tools and workflows, enhancing productivity and creativity while developing Magic Leap 2 applications. The SDK is supported by comprehensive documentation, sample projects, and an active developer community, which provide valuable resources and support to help developers quickly master the platform and overcome challenges during the development process. Overall, the Magic Leap SDK is a powerful and versatile toolset that empowers developers to create highly interactive and immersive AR applications.

3.1.4 MRTK 3

MRTK (Mixed Reality Toolkit) 3 [5] is an open-source framework designed to accelerate the development of MR applications across various platforms, including Microsoft HoloLens, Windows Mixed Reality headsets, and other AR and VR devices, such as Magic Leap 2. As an evolution of its predecessors, MRTK 2 and MRTK for Unity, MRTK 3 introduces significant improvements and new features to enhance the development process and the user experience. It serves as a comprehensive toolkit that provides developers with a wide range of components, scripts, and tools to create immersive and interactive MR experiences efficiently.

One of the key features of MRTK 3 is its support for multiple platforms, enabling developers to target a diverse range of devices and ecosystems. The framework abstracts the underlying platform-specific APIs and functionalities, allowing developers to write code once and deploy it across different MR platforms seamlessly. This cross-platform compatibility reduces development time and effort, enabling developers to reach a broader audience with their MR applications.

MRTK 3 offers a rich set of building blocks and prefabs that simplify common MR development tasks, such as hand tracking, spatial mapping, and gesture recognition. These components provide developers with the tools they need to create intuitive and immersive user interfaces and interactions in their MR applications. Additionally, MRTK 3 includes support for advanced features like spatial anchors, which enable persistent holograms in the physical environment, and spatial understanding, which allows applications to interact intelligently with the real world.

Another notable aspect of MRTK 3 is its emphasis on customization and extensibility. The framework provides developers with the flexibility to customize and extend its components to meet their specific requirements. This includes the ability to create custom gestures, add new input sources, and integrate third-party plugins and services seamlessly. Furthermore, MRTK 3's modular architecture allows developers to cherry-pick the components they need for their projects, reducing unnecessary overhead and improving performance.

Moreover, MRTK 3 benefits from a vibrant and active community of developers and contributors. The framework is open-source, allowing developers to contribute improvements, bug fixes, and new features back to the community. This collaborative approach fosters innovation and ensures that MRTK remains at the forefront of MR development. Additionally, MRTK provides extensive documentation, tutorials, and samples to help developers get started quickly and make the most of the framework's capabilities, making it an invaluable resource for MR developers of all skill levels.

3.1.5 Cesium Ion

Cesium Ion [6] is a powerful geospatial platform designed to streamline the creation, management, and visualization of **3D geospatial data** on the web. Developed by Cesium, Ion offers a suite of tools and services tailored for professionals in fields such as urban planning, architecture, defense, and telecommunications. At its core, Cesium Ion enables users to easily upload, process, and host large-scale 3D geospatial datasets, making it accessible for visualization and analysis in (but not limited to) web-based applications and more.

One of the key features of Cesium Ion is its ability to handle massive geospatial datasets with efficiency and scalability. The platform supports a wide range of geospatial data formats, including 3D models, point clouds, terrain, and imagery. Users can upload datasets of virtually any size and complexity, and Cesium Ion automatically processes and optimizes the data for web-based visualization. This enables users to create rich and immersive 3D maps and visualizations that can be accessed and shared easily via web browsers, without the need for specialized software or hardware.

Cesium Ion offers advanced tools for data management and collaboration, allowing users to organize, annotate, and share geospatial datasets with team members and end users. The platform provides version control and revision history features, enabling users to track changes and collaborate effectively on large-scale projects. Additionally, Cesium Ion's integration with popular geospatial tools and platforms, such as QGIS, ArcGIS, and Autodesk, facilitates seamless workflows and interoperability, ensuring that users can leverage their existing workflows and tools within the Cesium ecosystem.

Another notable aspect of Cesium Ion is its support for real-time streaming and dynamic visualization of geospatial data. The platform leverages the power of WebGL and modern web technologies to render 3D geospatial data in real-time, enabling users to visualize dynamic data sources, such as IoT sensor data, weather simulations, and **vehicle trajectories**, on interactive 3D maps. This capability opens up new possibilities for real-time monitoring, situational awareness, and decision-making in a wide range of applications, from emergency response and disaster management to asset tracking and urban planning.

Moreover, Cesium Ion offers a range of developer-friendly features and APIs that enable customization and extensibility. Developers can leverage CesiumJS, an open-source JavaScript library built on top of WebGL, to create custom web-based applications and visualizations using Cesium Ion's geospatial data services. Additionally, Cesium Ion provides APIs for integrating geospatial data into existing workflows and applications, enabling seamless integration with third-party systems and platforms, such as Unity 3D. This flexibility and openness make Cesium Ion a powerful and versatile platform for unlocking the potential of 3D geospatial data on the web.

3.1.6 Geospatial REST Services

In order to combine physical world operations and digital AR projections, a transformation of information and data needs to be performed. More specifically, HERON project relies on real world coordinates to identify locations, pinpoint maintenance sites and coordinate roadwork operations. The interconnecting channel(s) of all subsystems should carry, in most cases, geospatial references and geospatial data.

Conversions and projections take place through simple REST services, implemented to ensure data compatibility among different modules. One example is the **mapping** of GPS coordinates of the real world to a 3D space of the AR environment, as perceived from the operator of Magic Leap 2 devices. These projections are performed by geospatial and mapping libraries available in public channels and serve as the baseline of the aforementioned services. Common integration and deployment practices are then utilized for the continued integration / continued deployment needs of the project.

Finally, these services can optionally visualize data when accessed from a browser, further enhancing the situational awareness and providing a comprehensive view of the site.

3.2 Development Methodology

The development of an AR application for the Magic Leap 2 glasses involves a meticulous and structured methodology to ensure the final product meets user expectations, technical requirements, and industry standards. This sub-section outlines the development methodology employed in this project, detailing the phases, tools, and techniques used throughout the process.

The initial phase focused on understanding the project scope, defining requirements, and planning the development process. Key activities included:

- **Meetings**: Engaging with pilot site owners to gather requirements and understand user needs.
- **Feasibility Study**: Assessing the technical feasibility of the application on the Magic Leap 2 platform.
- **Requirement Documentation**: Creating detailed requirement specifications to guide the development process.

In the design phase, the application's architecture, UI, and interaction models were developed. Key activities included:

- **System Architecture Design**: Defining the core components of the application, including the AR Foundation engine / provider, UI, and data management systems.
- **UI/UX Design**: Crafting intuitive and user-friendly interfaces tailored for AR interactions using MRTK3.
- **Prototyping**: Developing low-fidelity prototypes to validate design concepts and gather early feedback.

The development phase involved coding and integrating the designed components into a functional application. Key activities included:

- **Environment Setup**: Setting up the development environment with necessary tools and SDKs, including Unity 3D, Magic Leap SDK, AR Foundation, Cesium Ion and other relevant software.
- **Feature Implementation**: Iteratively developing application features, such as object recognition, gesture control, and spatial mapping.
- **Integration**: Ensuring seamless integration of different components, including third-party libraries and APIs (e.g. geospatial services).

Early testing was conducted to identify and resolve issues, ensuring the application's reliability and performance. Key activities included:

- **Unit Testing**: Testing individual components for functionality.
- **Integration Testing**: Verifying that different components work together as expected.
- **User Testing**: Conducting usability testing with test users on the field to gather feedback and identify areas for improving the AR experience.
- **Performance Testing**: Assessing the application's performance, including frame rate and responsiveness, on the Magic Leap 2 hardware.

The final phase involved deploying the application and planning for ongoing improvements. Key activities included:

- **Deployment**: Releasing the application to the intended distribution channels.
- **Training / Support**: Providing documentation and training to intended users on the field.

The development of the AR application leveraged a variety of tools and technologies to facilitate efficient and effective development. Apart from the aforementioned hardware and software tools described in the previous section 3, other key tools included:

- **Git** [7]: A version control system to manage codebase changes and collaborate among team members.
- **JIRA** [8]: A project management tool used to track progress, manage tasks, and handle issues.

An Agile development approach was adopted to ensure flexibility and continuous improvement throughout the project. Key practices included:

- **Scrum Framework**: Implementing Scrum with regular sprints, sprint planning meetings, daily stand-ups, and sprint reviews.
- **Iterative Development**: Developing the application in iterations, allowing for incremental improvements and frequent reassessment.

The development methodology for the AR application using Magic Leap 2 glasses was designed to ensure a structured yet flexible approach, accommodating the unique challenges of AR development. By leveraging industry-standard tools, adhering to Agile practices, and maintaining a focus on user-centered design, the project aimed to deliver a high-quality, immersive AR experience.

4 AR User Interface (UI)

The UI of the customized AR application for Magic Leap 2 glasses is presented in this section. Apart from application's main menus, additional widgets and other visual aids are also described.

4.1 Application Menus

Throughout the application, users encounter various menus carefully designed to streamline navigation and enhance usability. The login screen serves as the initial point of entry, where users input their credentials using a virtual keyboard for seamless access.

Upon logging in, users are greeted with a welcoming menu offering clear and intuitive options. This main menu provides users with the choice to initiate a new calibration process or select a previously stored calibration, ensuring they can tailor the app precisely to their current conditions and requirements. Additionally, the main menu offers options to adjust the position of objects in the world, such as the observation map, allowing users to choose whether it will be placed on their left or right side.

Figure 2: Welcoming screen of AR application (testing in controlled environment).

Upon completing the calibration process, users can access a secondary interface using the corresponding left-hand gesture. This interface serves as the central hub for cone placement tasks, offering users essential options for efficient navigation. Within this interface, users can reset cone positions, adjust their placement, or finalize their decisions by sending the final path to the robotic device. These comprehensive options ensure users have the necessary tools to manage cone placement effectively, enhancing overall operational efficiency.

Figure 3: Left-hand gestures (testing in controlled environment).

Users may encounter additional menus throughout the application, each serving a specific purpose to assist users in their tasks. These menus are strategically placed to provide quick access to relevant features and options, ensuring a smooth and intuitive user experience.

Overall, the thoughtful integration of menus throughout the application enhances user navigation and facilitates task completion. By presenting users with clear and intuitive options at each step, the application ensures that users can efficiently access and utilize its features to meet their maintenance objectives.

4.2 Configurable Widgets List

Within the application’s UI, text and input fields serve as fundamental elements for user interaction and data entry. These fields play a pivotal role in facilitating the input of various types of information, such as usernames, passwords, or numerical values. Some of these text and input fields are sourced from the MRTK3, a comprehensive suite of tools and components tailored for crafting mixed reality experiences. MRTK-provided fields offer advanced functionality and seamless integration with mixed reality environments, ensuring optimal performance. Additionally, the application may utilize text and input fields native to the Unity 3D engine itself. These Unity-provided fields offer versatility and flexibility, allowing for customization of appearance and behavior to suit specific application requirements. By leveraging both MRTK and Unity-provided text and input fields, the application delivers a cohesive and intuitive user experience, enabling users to interact with the interface seamlessly and input data accurately.

Furthermore, the virtual keyboard provided by the Magic Leap SDK offers a visually appealing alternative to the default input method. Its sleek design and intuitive interface enhance the overall aesthetics of text entry, providing users with a more engaging and visually pleasing experience.

For the informational panels detailing robot specifications, advanced UI features such as sliders, video buttons, and dropdown lists are strategically employed to enhance user engagement and facilitate efficient data presentation. Sliders offer users intuitive control over numerical parameters, allowing for precise adjustment of settings related to the robotic device's specifications. Video buttons provide dynamic multimedia content, enabling users to access visual representations or instructional videos pertaining to the robot's functionalities. Additionally, dropdown lists offer users a structured selection interface, allowing for the categorization and easy retrieval of relevant information about the robot. By incorporating these advanced UI features into the information panel, the application ensures that users can access and interact with detailed robotic device specifications in a user-friendly and efficient manner, promoting enhanced comprehension and usability.

5 System Dependencies

In the following sub-sections, the dependencies and prerequisites for necessary AR components' implementation (as they have been finally refined and defined) will be presented. Integration and interaction between AR and other HERON components will also be described here.

5.1 Technical and Environmental Requirements

The technical and environmental requirements for implementing AR applications for Magic Leap 2 glasses using Unity 3D are presented below.

Hardware requirements:

- **Magic Leap 2 Device**: The primary AR headset for development and testing.
- **Development Computer**: A capable PC.
- **USB-C Cable**: For connecting the Magic Leap 2 to the development computer.
- **External GPS device**: For example, the user's smartphone or any other Bluetooth GPS localization device available. The selected device should be a handheld device. No additional requirement are imposed on position accuracy or update intervals. Of course, higher accuracy will result in an optimal user experience.

Software requirements:

- Operating System: Windows 10 or later, macOS 10.15 (Catalina) or later.
- Magic Leap Tools:
  - **Magic Leap Hub**: A desktop application for managing the Magic Leap device.
  - **MLSDK**: The software development kit for Magic Leap, which includes necessary libraries and tools.
- Development Environment:
  - **Unity**: Version 2020.3 LTS or later, for Unity-based development.
  - **Visual Studio**: Version 2019 or later, for coding in C# (Unity).
- **Magic Leap Unity SDK**: For Unity developers, includes plugins and APIs.
- Additional Software:
  - **AR Foundation**: For Unity developers, to create cross-platform AR experiences.
  - **XR Plugin Management**: For managing XR plugins within Unity.
- External software:
  - **External Bluetooth GPS app**: For receiving GPS location status through Bluetooth on the AR device. Existing freely available apps should suffice in terms of position accuracy. The target architecture should be the Android platform.

Environmental Requirements:

- Physical Space:
  - **Development Area**: A well-lit, open space with enough room to move around while wearing the Magic Leap 2 headset. Avoid clutter and ensure a safe environment to prevent accidents during testing.
  - **Testing Environment**: A variety of testing environments (outdoor, different lighting conditions, weather) to ensure the AR components perform well in diverse real-world conditions.
- Connectivity:

  - **Internet Connection**: Reliable and fast internet for downloading SDKs, tools, and updates, and for collaboration and cloud or other services.
  - **Wi-Fi**: Ensure strong and stable Wi-Fi connectivity, especially since the AR application involves networked experiences and real-time data streaming.
  - **Bluetooth**: A fast Bluetooth protocol is required for a fast and strong connection to peripheral devices, such as a Bluetooth GPS provider device, as well as the handheld Magic Leap Controller.
- Power and Battery Management:
  - **Chargers and Spare Batteries**: Ensure that chargers and spare batteries for the Magic Leap 2 and development devices are readily available to avoid downtime.

The technical (software) requirements for implementing AR applications for Magic Leap 2 glasses using MRTK3 are:

- **MRTK3 Foundation Package**: Core components of the Mixed Reality Toolkit.
- **MRTK3 Tools Package**: Additional tools and utilities for development.
- **MRTK3 Extensions Package**: Extensions for specific functionalities and integrations.

The technical requirements for implementing AR applications for Magic Leap 2 glasses using Cesium Ion services are:

- Cesium Ion Account:
  - **Cesium Ion**: Online account to access the platform's geospatial data services.
  - **API Key**: Generate an API key from the Cesium Ion dashboard for integrating Cesium services into an AR application.
- Cesium Plugins:
  - **Cesium for Unity**: For Unity integration.

Additional technical requirements are imposed by Google Maps Static API, that is utilized both by Cesium Ion and our AR components, when a static map tile is required. These requirements include:

- Google Cloud Platform:
  - **Google Maps API**: Google Cloud account and activation of the Google Maps Static API.
  - **API Key**: Generate an API key from the Google Cloud Console for integrating Google Maps Static API services into an AR application.

Finally, the technical and environmental requirements for implementing Geospatial REST services for supporting an AR application's needs are presented below.
Hardware requirements:

- **Server** (for deployment): Basic CPU/Memory/Storage/Network requirements.

Software requirements:

- Development Environment:
  - **Python**: Version 3.7 or later.
  - **FastAPI**: Latest version.
  - **Uvicorn**: ASGI server for running FastAPI applications.
  - **Database**: PostgreSQL with PostGIS extension, or another geospatial database (e.g., Spatialite).
  - **Geospatial Libraries**: GeoPandas
- Geospatial Services:

  - **Service Provider**: A geospatial data provider, in our case Google Maps Static API.
  - **API Key or Token**: Necessary authentication credentials from the service provider to access geospatial data. In our case, Google Maps API Key.
- **Containerization**: Docker for containerizing applications.
- API and Data Handling:
  - **Requests**: Library for making HTTP requests.
  - **Pydantic**: For data validation and serialization.
  - **SQLAlchemy**: For database ORM.
  - **Alembic**: For database migrations.

Environmental requirements:

- Server Environment/Deployment:
  - **On-Premises, Data Center or Cloud Environment.**

5.2 User Readiness

The state of preparedness of end-users to interact with the system is discussed in this sub-section. This readiness encompasses several aspects related to the system's dependencies, as follows:

- **Technical Proficiency**: Users should have basic understanding of AR display principles and basic familiarization of digital overlays over real-world objects. This should provide a good understanding of how to navigate the user interface, perform required actions, and troubleshoot common issues. Users can further develop a better understanding through training sessions, tutorials, or user manuals provided by the development team.
- **Access to Resources**: Users must have access to the resources required by the system to function properly. These include hardware resources, which are Magic Leap 2 glasses, a Bluetooth GPS device (e.g. smartphone) and a Wi-Fi access point on site with internet connection. Software dependencies include the HERON AR application that runs on Magic Leap 2 glasses and a (free) external Bluetooth GPS receiver application, that mocks the device's location (in our case, the AR device). In the case of the external GPS device being an Android device, (free) Bluetooth GPS transmitter applications exist in the respective stores. Ensuring that users have access to the above resources is essential for their ability to use the system without impediments.
- **Data Availability**: Users should have access to the data required by the system to perform their tasks. Access to Geospatial REST services, respective databases and files, as well as Geospatial APIs is ensured through a stable internet connection. Access to external APIs that the system relies on for data processing or retrieval, such as Cesium Ion servers and Google Maps Tile servers, is also ensured by a stable internet connection. Data provisioning efforts by the development team to ensure that users have access to the necessary data sources is not necessary, as access rights are granted through the use of our AR application.
- **Understanding of Functionalities**: Understanding the purpose of different features, how to initiate specific actions, and the expected outcomes of the interactions with the system are part of the training material provided by the development team. Additionally, clear documentation, training materials and user support channels should be provided, along side the live demo of the operation of the application.
- **Adaptability to Changes**: Support and resources to navigate any software changes effectively should include communication about upcoming changes, training on new

features, or assistance with troubleshooting compatibility issues. Communicating changes to end-users on the field in crucial to the correct operation and deployment of new software.

5.3 Communication Frame

A structured approach for facilitating communication between different stakeholders regarding system dependencies is essential for HERON project. This framework helps ensure that all relevant parties are informed, aligned, and able to collaborate effectively throughout the development process. Specifically:

- **Stakeholder Identification**: Stakeholders involved in the system dependencies include developers, testers, end-users and system administrators.
- **Communication Channels**: Established meetings, emails, project management tools and dedicated collaboration platforms have been established so that information flows smoothly and reaches the intended recipients in a timely manner.
- **Information Sharing**: Updates on software versions, changes to hardware requirements and availability of resources are communicated through regular status updates, release notes, or documentation updates.
- **Timing and Frequency**: When required, this involves scheduling regular meetings, sending periodic updates, or providing real-time notifications for urgent matters. It's at the discretion of each stakeholder to adjust the timing and frequency based on the priority of each issue.
- **Responsiveness and Feedback Mechanisms**: Mechanisms for stakeholders to raise questions, report issues, or request clarification about system dependencies should follow the already established channels' communication guidelines. Additionally, live demos should give the opportunity for live feedback and interactive response from end-users on the field. This promotes transparency, collaboration, and continuous improvement throughout the development process.
- **Conflict Resolution**: Addressing conflicts in a timely and constructive manner helps maintain harmony and progress within the project team.
- **Documentation and Record-Keeping**: It is of paramount importance to maintain thorough documentation of all communication related to system dependencies, including meeting minutes, email correspondence, decision logs, and updated documentation. This serves as a reference for stakeholders and helps ensure that everyone is working from the same information.
- **Continuous Improvement**: Solicit feedback from stakeholders, especially end-users, on their communication needs and preferences is crucial, as well as the ability to incorporate lessons learned from past experiences to refine the communication processes over time.

6 Implementation of AR Components

6.1 Challenges and Limitations

The HERON project utilizes geospatial information, including latitude, longitude, and height, of various entities involved in the operational procedure. The precise geographical positions of the cones must be accurately captured and properly visualized within the world space. Additionally, the position of the robot must be known at all times during the application's runtime, as well as the path it has traversed to complete the cone placement task and perform the necessary road maintenance.

Furthermore, the current location of the end-user or operator on the field, is of critical importance, as it serves as the origin point within the AR application space. This information is essential for ensuring that all spatial relationships and movements are accurately represented and coordinated. The integration and precise tracking of these geospatial data points are fundamental to the successful execution of the HERON project's objectives, necessitating a high level of accuracy and reliability in the geospatial information utilized.

In addition to the inherent challenges associated with the geospatial projection to world space, there are several critical hardware-related issues and limitations that need to be meticulously addressed. Current commercial-use GPS sensors, such as those commonly integrated into smartphones, exhibit a notable limitation in their positional accuracy. These devices typically present a positional offset that exceeds 3 meters between the true geographic location and the coordinates reported by the sensor.

The HERON use case demands a level of accuracy that current GPS technology cannot reliably provide. Specifically, the acceptable margin of error for the placement of cones is in the range of centimetres. Any deviation beyond this narrow threshold can lead to substantial discrepancies, thereby compromising the effectiveness and reliability of the application. Therefore, the current state of GPS technology, with its inherent limitations, falls short of meeting the minimum precision requirements for HERON's smooth operation (as reported by early testing on real road conditions).

Addressing these hardware limitations is crucial for ensuring the successful implementation of the HERON project. This may involve exploring alternative technologies or augmenting existing GPS systems with additional correction mechanisms to achieve the desired level of accuracy. Such enhancements are essential to bridge the gap between the current technological capabilities of handheld GPS devices (e.g. smartphones) and the precise requirements of the HERON's use case(s).

6.2 Calibration Process

For the integration of AR components into the world space, a meticulous calibration process is indispensable. This procedure ensures the precise mapping of the Unity world space to real-world geographical coordinates.

Initially, the calibration process commences with the selection of a calibration point. At this stage, the application interacts with various services to facilitate the calibration. The first step involves capturing a satellite image of the designated area. This image is then passed onto a set of geospatial REST services, responsible for analyzing the images and performing the necessary transformations. This aims to identify prominent features, particularly the edges of traffic lanes / road lanes.

Sophisticated computer vision algorithms are deployed within these services to meticulously detect the edges of road lanes from the captured satellite image. The appropriate image transformations, including edge detection algorithms and filtering techniques, are applied to enhance the visibility of traffic lane features. These transformations help highlight the edges by accentuating changes in pixel intensity and gradient across the image.

After applying the necessary transformations, the computer vision algorithms meticulously analyze the processed image to identify potential road lane edges. Each detected edge undergoes rigorous evaluation based on several criteria, including confidence levels, slope values, and continuity along the road geometry.

After all the road line edges are detected and evaluated, the system selects the most suitable edge for calibration. This selection process involves assessing various parameters, including the confidence score of each detected edge, the consistency of its slope values, and other relevant factors. The edge with the highest score, indicating greater confidence in its accuracy and alignment with the actual road geometry, may be chosen as the calibration point (although the selection is not mandatory).

This approach ensures that the calibration point selected for aligning the virtual environment with the real-world space is robust and reliable. By leveraging advanced computer vision techniques and rigorous evaluation criteria, the system optimizes the accuracy and effectiveness of the calibration process, laying the foundation for seamless integration of AR components into HERON’s AR application.

Following the selection of the calibration point, the user is required to physically relocate and stand at this designated point in the real world. To facilitate this crucial step, the application provides the user with a satellite map of the area in question. This satellite map, integrated into the UI, displays markers indicating both the selected calibration point and the user’s current location in real time.

By presenting this visual representation of the calibration process, the user is aided in accurately navigating to the chosen calibration point. The markers on the satellite map dynamically update to reflect any movements made by the user, ensuring real-time alignment between the virtual representation and the physical environment. This visual feedback enhances the user experience and facilitates the seamless completion of the calibration process, laying the groundwork for the integration of AR components into HERON’s AR application with accuracy and efficiency.

6.3 AR Objects

Expanding upon the implementation of AR objects within the HERON project’s framework, it’s essential to delve into the intricate processes and benefits associated with this technology integration.

The utilization of Unity space as the foundation for AR component implementation offers several distinct advantages. One notable advantage lies in its ability to faithfully mirror real-world space on a one-to-one scale. This synchronization is pivotal, as it enables seamless integration between the virtual and physical environments. By aligning virtual objects with their real-world counterparts, users experience a heightened sense of immersion and realism within the AR application.

Figure 4: Overview of integration between the virtual and physical environments (testing in controlled environment).

Integral to this synchronization is the application of a geo-projection method. This method serves as the bridge between virtual AR elements and their respective geographical coordinates in the real world. Through precise geo-projection, AR cones are accurately overlaid onto their designated locations, ensuring a seamless fusion between the virtual and physical realms.

The meticulous implementation of the geo-projection method is paramount to achieving precise alignment between virtual AR elements and their real-world counterparts. This precision is essential for ensuring that users can interact with AR cones with exactitude within HERON's AR environment. Whether it's for cone placement tasks or accessing relevant information about the robot's position in real time, users can rely on the accuracy and fidelity of the AR functionality.

Furthermore, the ability to view real-time information about the robot above its exact world position adds another layer of functionality and utility to the application. This feature not only enhances the spatial visualization capabilities, but also contributes to the overall efficiency and effectiveness of HERON-related processes and maintenance operations.

Figure 5: Overlay of AR cones onto designated locations (testing in controlled environment).

Moreover, the SDK furnished by the Magic Leap 2 glasses presents sophisticated features, notably hand tracking. This functionality empowers users to engage with AR objects through intuitive hand gestures. By discerning multiple hand key points and identifying specific gestures, users seamlessly manipulate virtual objects within the AR environment. For instance, users can execute tasks, such as object manipulation, by employing finger pinching gestures or access supplementary information with a mere pointing gesture.

6.4 Cone Position Adjustment

Following the initial cone coordinates received by the Sensing Interface and AI component (through the Middleware), an additional step is seamlessly integrated into the process, where a tabletop satellite map of the area becomes instrumental. This feature leverages Google Maps Static API to procure a detailed satellite image of the designated area. Subsequently, this satellite image is projected onto a tabletop-style circular map within the application.

This strategic fusion of satellite imagery with AR technology empowers users to engage with spatial data in a comprehensive manner. The circular map serves as a tangible representation of the geographical context, providing users with a visual reference for the placement of AR cones.

Furthermore, AR cones are overlaid as three-dimensional objects atop the satellite map. This overlay allows users to precisely adjust the placement of cones according to their specific requirements. By selecting individual cones on the circular map, users can manipulate their positions with ease, ensuring optimal placement for the intended tasks.

Moreover, the integration of zoom functionality enhances the precision of cone positioning. Users have the ability to zoom-in and out on each cone, enabling them to fine-tune their positions with increased accuracy. This capability is particularly valuable in scenarios where meticulous placement of cones is paramount, such as in road maintenance tasks or traffic management operations.

In addition to the placement of AR cones on the tabletop satellite map, another key feature that enhances spatial visualization within the application is the following: Each AR cone is not only positioned individually, but is also interconnected with lines, visually representing the polygon they collectively create. This polygon delineates the operational area within which the robot works, providing users with a clear visual representation of the spatial extent covered by the cones and the boundaries of the designated work area. By connecting the cones with lines to form this polygon, users can easily assess the coverage area and ensure efficient and safe deployment of resources during related tasks and operations.

Figure 6: Example of positioned AR cones (orange circles) and the relevant operational area (polygon delimited by the orange dash interconnection lines among the cones) on tabletop satellite map.

6.5 Area Overview and Situational Awareness

Within the project, a satellite 3D map facing the operator at all times serves a dual purpose; notably for the observation of the surrounding area and the entire operation procedure. This advanced mapping tool utilizes the Cesium World Globe, providing a comprehensive view of the operational landscape while conserving computational resources.

Through the utilization of Cesium 3D tiles technology, the satellite imagery offers detailed and accurate representations of the operational area. By dynamically loading only the tiles corresponding to the area of interest, the application optimizes performance and minimizes data usage. This approach ensures that operators have access to high-fidelity imagery without compromising on efficiency.

The map precisely depicts the location of the operator, the cone(s) polygon, and the robot's location along with the path it has covered since the start of the operation, further enhancing operational insight and coordination.

Apart from cone placement, the availability of a satellite 3D map purely for overview purposes (throughout the operation) is crucial. This tool enables operators to maintain continuous awareness of their surroundings and monitor ongoing activities in real time. Users can assess terrain features (e.g. slope), identify obstacles, and anticipate potential challenges, thereby enhancing overall operational effectiveness and safety.

The application allows for visualization of both the robot's current position and its complete trajectory from the operation's outset. By utilizing historical data and real-time tracking, users can access vital metrics concerning the robot's movement. This encompasses viewing the initial position, destination, and intermediate waypoints along the route. Furthermore, users have the capability to measure the total distance travelled by the robot since the operation's commencement.

Figure 7: Example of visualization of the robot's position (blue rectangle) and trajectory (blue dash line) on the satellite map.

This comprehensive view of the robot's path enhances operational insight and coordination by providing users with valuable information about the robotic device's movements and progress. It enables operators to monitor the efficiency of the robot's navigation, identify potential areas for optimization, and make informed decisions to ensure the successful and safe execution of the operation.

Moreover, users have the capability to select each of the cones displayed on the 3D map and access additional content about their geospatial information. This feature enhances the user's ability to gather detailed insights relevant to specific cones, facilitating better understanding and informed decision-making within the project's operational context.

This multifaceted use of 3D maps underscores their pivotal role in enhancing situational awareness and operational effectiveness within the project. By providing operators with a comprehensive view of the operational landscape and facilitating access to critical information, the satellite 3D map significantly contributes to the success and efficiency of project operations.

Figure 8: Example of cones selection on the satellite map, in order to display their geospatial information.

7 Integration with other HERON Components

In the line of HERON project's scope, as it was noted in the section 2 of this deliverable and covered throughout the text in the deliverable D2.2 [1], the AR components were designed to integrate/interact with some other sub-systems and components within the HERON specified architecture. The following list compiles those components that present a direct or indirect interrelation:

- **Sensing Interface and AI**: the server-hosted Computer Vision system determines the locations of cone placement (a list of coordinates for where the cones should be placed), which the road authority's field crew member supervising the maintenance operational mission can see on the AR glasses to intervene if necessary.
- **Robotic system (UGV and HLP)**: supports live (video) streaming and information overlay compatibility with AR components.
- **Middleware and DF**: provides the AR components with the aforementioned cone placement data, as well as robot sensors' telemetry data, and other digital assets on the maintenance field.
- **Secure Data Communication**: enables the data flow among AR and the other HERON components, through the established (Kafka broker) architecture.
- **IMS/COP platform**: integrated with AR-based visualization capabilities. AR app offers an intuitive 3D map engine for operational planning and implementation. By the integration, an enhanced real-time visualization is supported under the advanced mapping environment of IMS/COP platform.

The specified interaction of AR device with the above HERON components under the road maintenance operations is presented by a workflow below (see Fig. 9). Any updates on the integrated components can possibly impact the AR functionality.

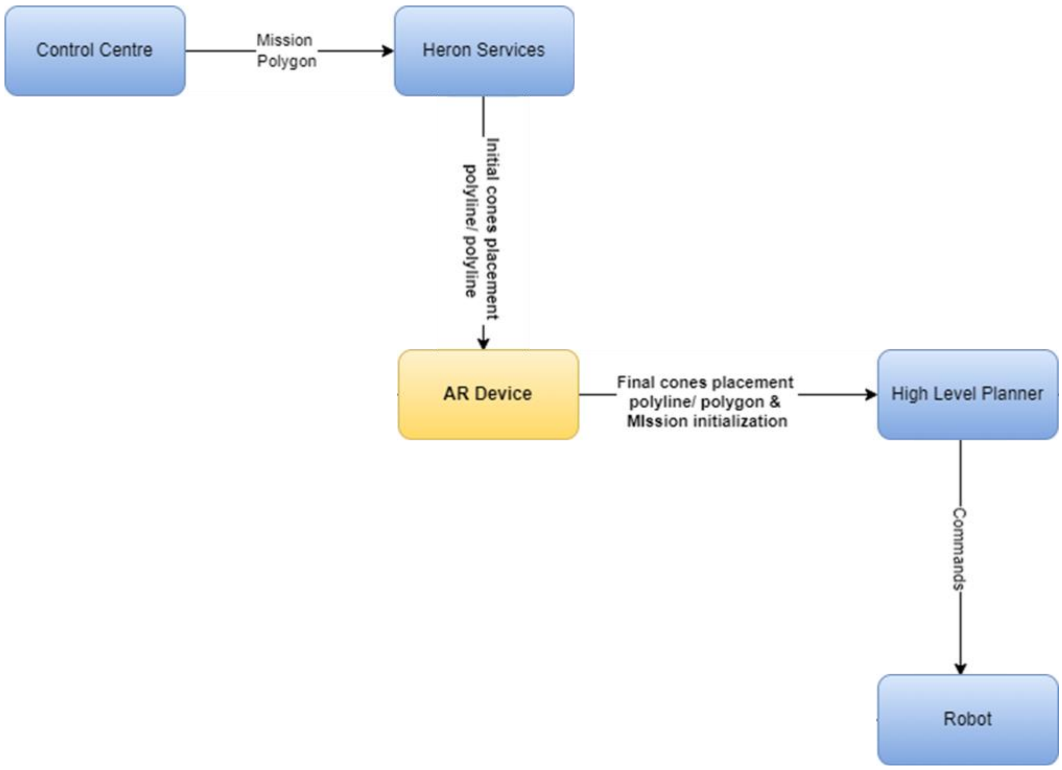

Figure 9: AR component in maintenance operations workflow.

8 Conclusions

In conclusion, this document has provided a comprehensive overview of the technological components developed under the HERON framework, aiming at RI maintenance supervisors and/or robot operators in order to monitor and manage their field road operations with the help of AR technology. Key aspects such as the specifications of hardware and software components, their system dependencies and prerequisites, and their implementation for road operational monitoring and management, are highlighted.

The AR components as standalone system provides a detailed and enriched view of the maintenance site, allowing for better decision-making and oversight. Through the AR glasses, the involved users can visualize real-time information and data onto the physical RI environment. Through the AR interface, they can observe the robot's operations, such as the cone placement and position adjustment, and make real-time adjustments or provide additional instructions if needed. The provided operational awareness is more enhanced by the integration of AR system with other HERON (sub)systems.

Despite the obvious advantages, the AR system should be properly developed and configured within the HERON project. Its detailed design was based on both the functional requirements and implementation scope. The result of the development mainly consists of an Android application for AR glasses that is user-friendly. However, the users must take into consideration some of the following aspects before using it:

- Understanding the specific background for AR technology and systems.
- Understanding the AR system's scope.
- Understanding the system's features and functions.
- Accommodating with the UI specifically designed for the HERON framework.
- Following specific technical guidelines and recommendations.
- Connecting and integrating the system within the HERON specified architecture of (sub)systems.

References

[1] D2.2: Architecture Specification. HERON project – 955356
[2] Magic Leap 2. https://www.magicleap.com/magic-leap-2 (accessed 21/05/2024)
[3] Unity 3D Engine. https://docs.unity3d.com/Manual/index.html (accessed 21/05/2024)
[4] AR Foundation - ARKit - ARCore https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@6.0/manual/index.html (accessed 21/05/2024)
[5] MRTK 3. https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk3-overview/ (accessed 21/05/2024)
[6] Cesium Ion. https://cesium.com/platform/cesium-ion/ (accessed 21/05/2024)
[7] Git. https://www.git-scm.com/ (accessed 22/05/2024)
[8] JIRA. https://www.atlassian.com/software/jira (accessed 22/05/2024)