



VRIJE
UNIVERSITEIT
BRUSSEL



MASTER STUDY

Real-Time Remote AI Segmentation over
4G/5G Networks for Assistive Navigation
on Custom Trails for Visually Impaired
Persons (VIPs)

Maarten Dequanter

September 4, 2026

Faculteit ingenieurswetenschappen

Contents

1	Introduction	7
1.1	Context	7
1.2	Objectives and contributions	7
1.2.1	Analyse Edge versus Cloud AI Performance for Real-Time blind Assistive trail path Navigation	7
1.2.2	Define methodology for Safe and Repeatable Evaluation	7
1.2.3	Design proof of concept: 5G Trail path navigation for VIPs	7
1.3	Thesis Structure	8
2	Related Work	9
2.1	Edge inference	9
2.2	Cloud inference	9
2.3	Overview	10
3	Preliminaries	11
3.1	Positioning and Mapping	11
3.2	Networking and Communication	11
3.3	Perception and AI Models	15
3.4	Feedback and Interaction	15
3.5	End-to-end Encryption via AES-256	17
3.6	Unreal Engine 5.6 as a Realistic Simulation Environment	17
4	Thesis Architecture set-up	18
4.1	Network architecture	18
4.1.1	Domestic 4/5G network route	18
4.1.2	Local Wi-Fi network route	18
4.1.3	Segmentation AI Inference Server (remote/cloud GPU)	18
4.1.4	Local Signaling Server	19
4.1.5	Netgear Nighthawk M6 5G router	19
4.2	Edge-devices	20
4.2.1	Raspberry Pi 4 / 5	20
4.2.2	Standard laptop	20
4.2.3	Smartphone (Stay on trails app)	21
5	Methods	22
5.1	Experimental setup	22
5.1.1	Digital twin and FPV data generation	22
5.1.2	Segmentation model for path detection	22
5.1.3	Reproducible streaming playback	24
5.2	Measurement design	25
5.2.1	Standard 5G metrics in related research	25
5.2.2	Additional metrics introduced in this thesis	25
5.2.3	Summary of logged fields	25
5.3	Feedback visualization	25
5.3.1	Visualisation of haptic feedback	25
5.4	Benchmark setups	26
6	Benchmark measurements	28
6.1	Real-time inference via an external GPU	28
6.1.1	Real-time responsiveness benchmarks (Latency, FPS and bitrate)	28
6.1.2	Conclusions	30
6.1.3	Additional metrics	30
6.1.4	Conclusions	34
6.2	Real-time local inference on edge-devices	35
6.2.1	Standard laptop	35
6.2.2	RPI4 and RPI5	35
6.3	Stability of latency and Link Quality over 4/5G Network Setup	37
6.3.1	Objectives	37
6.3.2	Edge-device hardware setup (RPI5)	37
6.3.3	Measurements	37
6.3.4	Conclusions	38
7	Conclusions	39

8	Ongoing developments and Future work	40
8.1	Proof-of-Concept: Real-Time Interactive Simulation of a VIP in Unreal Engine	40
8.2	Smartphone Prototype: <i>Stay on Trails</i>	41
8.3	BlindNavpark(Koning Boudewijnstichting funding request)	43
8.4	Future Work	43
9	Word list	44
10	Appendix	45
10.1	TCP - Transport layer protocol	45
10.2	Websocket application layer protocol	45
10.3	WebRTC	45
10.4	Network Address Translation (NAT)	45
11	Bibliography	46

Abstract

Visually impaired persons (VIPs) face persistent challenges in safe outdoor navigation, particularly on unstructured trails where conventional aids such as white canes or smart canes fail to detect low obstacles or maintain consistent path following. This thesis investigates the feasibility of a real-time assistive navigation system that leverages computer vision, edge/cloud inference, and 5G mobile networks to provide reliable path guidance on custom trails in natural environments.

The research addresses three objectives: (1) benchmarking edge versus cloud AI inference for trail segmentation, (2) defining a safe and repeatable methodology for evaluating end-to-end performance, and (3) designing a proof-of-concept 5G-enabled navigation framework for VIPs.

A digital twin of a park environment, developed in Unreal Engine 5.6, was combined with a first-person view (FPV) camera setup that emulates a walking user. Pre-recorded FPV sequences ensured reproducibility across experiments. The system captures video frames on edge devices (Raspberry Pi (RPI) 4/5 and laptops), compresses and encrypts them, and transmits them via WebSocket over 4G/5G networks to a GPU inference server for segmentation.

Results demonstrate that cloud-based GPU inference over public 5G NR-NSA networks can maintain end-to-end latencies below 200 ms at acceptable framerates less than 20, which is well within the safe limits for real-time haptic feedback and walking guidance. Edge segmentation inference on low-power devices such as the RPI5 achieved only 1.9 FPS, confirming that cloud offloading remains essential for responsiveness. Field measurements in a Brussels park showed an average 5G latency of 24 ms and an average throughput of approximately 20 Mbps, supporting the feasibility of real-world deployment on a publicly available commercial 5G network. Power measurements further revealed that low-power edge devices consume only 2.5-3.5W while using external inference, enabling several hours of mobile operation.

Lessons learned include the importance of adaptive bitrate control to maintain frame-level reliability under fluctuating link quality, queue monitoring as an early warning indicator for overload, and the role of local failsafe mechanisms (e.g., smart cane or depth sensors) to ensure safety during temporary network degradation. The developed proof-of-concept robot simulation in Unreal Engine also proved valuable as a training and evaluation tool for VIP participants in a safe, controlled environment.

Overall, the thesis confirms that real-time cloud-assisted segmentation for blind navigation over public 5G networks is technically feasible, energy-efficient, and economically viable, paving the way for future large-scale implementations such as the **BlindNavParks** project funded by the Koning Baudewijnstichting.

Samenvatting

Visueel beperkte personen (VIP's) worden geconfronteerd met aanhoudende uitdagingen bij veilige buitennavigatie, in het bijzonder op ongestructureerde paden waar conventionele hulpmiddelen zoals de witte stok of slimme wandelstokken tekortschieten in het detecteren van lage obstakels of het consistent volgen van een pad. Deze scriptie onderzoekt de haalbaarheid van een real-time assistief navigatiesysteem dat gebruikmaakt van computer vision, edge- en cloudgebaseerde inferentie en mobiele 5G-netwerken om betrouwbare padbegeleiding te bieden op aangepaste paden in natuurlijke omgevingen.

Het onderzoek richt zich op drie doelstellingen: (1) het benchmarken van edge- versus cloudgebaseerde AI-inferentie voor padsegmentatie, (2) het definiëren van een veilige en reproduceerbare methodologie voor het evalueren van end-to-end prestaties, en (3) het ontwerpen van een proof-of-concept van een 5G-ondersteund navigatiekader voor VIP's.

Een digitale tweeling van een parkomgeving, ontwikkeld in Unreal Engine 5.6, werd gecombineerd met een first-person view (FPV)-cameraopstelling die een wandelende gebruiker simuleert. Vooraf opgenomen FPV-sequenties zorgden voor reproduceerbaarheid tussen experimenten. Het systeem capteert videoframes op edge-devices (Raspberry Pi (RPI) 4/5 en laptops), comprimeert en versleutelt deze, en verzendt ze via WebSocket over 4G/5G-netwerken naar een GPU-inferentieserver voor segmentatie.

Resultaten tonen aan dat cloudgebaseerde GPU-inferentie over publieke 5G NR-NSA-netwerken end-to-end latencies onder 200 ms kan aanhouden bij aanvaardbare framerates lager dan 20, wat ruimschoots binnen de veilige grenzen ligt voor real-time haptische feedback en begeleiding aan wandeltempo. Edgegebaseerde segmentatie-inferentie op energiezuinige toestellen zoals de RPI5 behaalde slechts 1.9 FPS, wat bevestigt dat cloud-offloading essentieel blijft voor voldoende responsiviteit. Terreinmetingen in een park in Brussel toonden een gemiddelde 5G-latency van 24 ms en een gemiddelde throughput van ongeveer 20 Mbps, wat de haalbaarheid van real-world inzet op een publiek beschikbaar commercieel 5G-netwerk ondersteunt. Vermogensmetingen tonen bovendien aan dat energiezuinige edge-devices slechts 2.5-3.5 W verbruiken bij gebruik van externe inferentie, wat meerdere uren mobiele werking mogelijk maakt.

Geleerde lessen omvatten het belang van adaptieve bitratecontrole om framebetrouwbaarheid te behouden bij fluctuerende linkkwaliteit, het monitoren van wachtrijen (queues) als vroege indicator voor overbelasting,

en de rol van lokale failsafe-mechanismen (zoals een slimme wandelstok of dieptesensoren) om veiligheid te garanderen tijdens tijdelijke netwerkdegradatie. De ontwikkelde proof-of-concept robotsimulatie in Unreal Engine bleek bovendien waardevol als trainings- en evaluatie-instrument voor VIP-deelnemers binnen een veilige, gecontroleerde omgeving.

Samenvattend bevestigt deze scriptie dat real-time cloudondersteunde segmentatie voor navigatie van blinden over publieke 5G-netwerken technisch haalbaar, energie-efficiënt en economisch levensvatbaar is. Dit effent het pad voor toekomstige grootschalige implementaties, zoals het **BlindNavParks**-project, gefinancierd door de Koning Boudewijnstichting.

Résumé

Les personnes malvoyantes ou aveugles (PMA) sont confrontées à des difficultés persistantes en matière de navigation extérieure sûre, en particulier sur des sentiers non structurés où les aides conventionnelles telles que la canne blanche ou les cannes intelligentes ne parviennent ni à détecter les obstacles bas ni à assurer un suivi fiable du chemin. Ce mémoire étudie la faisabilité d'un système de navigation assistée en temps réel exploitant la vision par ordinateur, l'inférence IA en périphérie et dans le cloud, ainsi que les réseaux mobiles 5G, afin de fournir un guidage fiable sur des parcours personnalisés en environnements naturels.

La recherche poursuit trois objectifs principaux : (1) comparer les performances de l'inférence IA en périphérie et dans le cloud pour la segmentation de sentiers, (2) définir une méthodologie sûre et reproductible pour l'évaluation des performances de bout en bout, et (3) concevoir un proof-of-concept d'un cadre de navigation assistée par la 5G destiné aux PMA.

Un jumeau numérique d'un environnement de parc, développé sous Unreal Engine 5.6, a été combiné à une configuration caméra en vue à la première personne (FPV) simulant un utilisateur en marche. Des séquences FPV préenregistrées ont permis d'assurer la reproductibilité des expériences. Le système capture les images vidéo sur des dispositifs en périphérie (Raspberry Pi (RPI) 4/5 et ordinateurs portables), les compresse et les chiffre, puis les transmet via WebSocket sur des réseaux 4G/5G vers un serveur d'inférence GPU chargé de la segmentation.

Résultats montrent que l'inférence GPU basée sur le cloud via des réseaux publics 5G NR-NSA peut maintenir des latences de bout en bout inférieures à 200 ms pour des fréquences d'images acceptables inférieures à 20, ce qui se situe largement dans les limites de sécurité requises pour un retour haptique en temps réel et un guidage à vitesse de marche. L'inférence de segmentation en périphérie sur des dispositifs à faible consommation, tels que le RPI5, n'a atteint que 1.9 FPS, confirmant que le recours au cloud reste indispensable pour garantir une réactivité suffisante. Des mesures de terrain réalisées dans un parc bruxellois ont révélé une latence moyenne 5G de 24 ms et un débit moyen d'environ 20 Mbps, confirmant la faisabilité d'un déploiement réel sur un réseau 5G commercial public. Les mesures de consommation ont en outre montré que les dispositifs en périphérie à faible consommation n'utilisent que 2.5 à 3.5 W lorsqu'ils recourent à une inférence externe, permettant plusieurs heures de fonctionnement mobile.

Leçons apprises incluent l'importance du contrôle adaptatif du débit afin de maintenir la fiabilité au niveau des images malgré les fluctuations de la qualité du lien, la surveillance des files d'attente (queues) comme indicateur précoce de surcharge, ainsi que le rôle des mécanismes de sécurité locaux (par exemple une canne intelligente ou des capteurs de profondeur) pour garantir la sécurité en cas de dégradation temporaire du réseau. Le proof-of-concept de simulation robotique développé sous Unreal Engine s'est également révélé précieux comme outil de formation et d'évaluation pour des participants PMA dans un environnement sûr et contrôlé.

Dans l'ensemble, ce mémoire confirme que la segmentation assistée par le cloud en temps réel pour la navigation des personnes aveugles via des réseaux publics 5G est techniquement réalisable, économe en énergie et économiquement viable. Il ouvre la voie à de futures implémentations à grande échelle, telles que le projet **BlindNavParks**, financé par la Fondation Roi Baudouin.

Acknowledgements

I would like to express my deepest gratitude to my wife An and my children Hendrik, Lies, and Willem for their patience and understanding during the past two to three years, while I was studying and working at the same time. Their support made this journey possible.

I also want to thank Maxime Frooninckx, a visually impaired person, for providing essential feedback to establish a foundation of the real needs of blind users. His input ensured that this research was not only technically sound but also relevant to the target community.

Special thanks go to Professor Jan Lemeire, the promotor of this thesis, whose valuable advice and attention points greatly contributed to shaping this work into its final form.

I would also like to thank Erasmushogeschool Brussel and its Research Center for Preventive Health Innovation for their support and willingness to make this project possible, as well as their active role in securing funding from the Koning Boudewijnstichting. Their encouragement was essential to the realization of this research.

A heartfelt thank you goes to our dog Snoozy, whose running trips in the Laerbeek forest were indispensable for gathering training data and for keeping me balanced during this process. Snoozy passed away in August 2025.

Finally, I wish to thank the Vrije Universiteit Brussel (VUB) and the Department of Industrial Sciences, including all professors and assistants, for the opportunity to follow the courses through their bridging programme. Their teaching and guidance provided the academic foundation needed to complete this master's degree.

Disclosure on the use of AI

During the preparation of this work, the author(s) used OpenAI ChatGPT: Assisted in drafting, restructuring, and refining thesis text generated example code snippets and table/figure scaffolds; checked terminology consistency and provided concise English translations of Dutch passages. After using these tool(s) and/or service(s), the author reviewed and edited the content as needed. The author takes full responsibility for the content of this work.

Abbreviations

- 5G NSA / SA = Non-Standalone vs. Standalone 5G; NSA uses 4G infrastructure with 5G radio, while SA has a dedicated 5G core, offering lower latency.
- ABR = Adaptive Bitrate; a streaming technique that dynamically adjusts video quality depending on network conditions.
- AES = Advanced Encryption Standard; a symmetric encryption algorithm used for secure data transmission.
- AI = Artificial Intelligence; the simulation of human intelligence by machines for perception, reasoning, and decision-making.
- CBC = Cipher Block Chaining; a mode of operation for block ciphers that provides confidentiality by linking each block to the previous one.
- CSV = Comma-Separated Values; a simple text format for storing tabular data.
- FPS = Frames per Second; the rate at which video frames are captured, processed, and transmitted in real time.
- FPV = First-Person View; a camera perspective simulating what the user or robot “sees” during navigation.
- GNSS = Global Navigation Satellite System; includes constellations such as GPS (United States) and Galileo (European Union).
- GPU = Graphics Processing Unit; a processor specialized in parallel computation for rendering and AI inference.
- IMU = Inertial Measurement Unit; measures acceleration, angular velocity, and magnetic field strength for motion estimation.
- JSON = JavaScript Object Notation; a lightweight text-based data format for exchanging structured information.
- mAP = Mean Average Precision; an evaluation metric for object detection and segmentation models.
- mIoU = Mean Intersection over Union; a metric evaluating segmentation accuracy by measuring overlap between predicted and ground-truth regions.
- NAT = Network Address Translation; a method allowing devices in a private local network to share a single public IP address.
- POC = Proof of Concept; an initial prototype used to demonstrate technical feasibility.
- QoS = Quality of Service; a measure of network performance including latency, jitter, and throughput.

- RGB = Red-Green-Blue; a color model used in digital imaging and computer vision.
- RPI = Raspberry Pi; a family of small, single-board computers used as edge devices in this study.
- RTT = Round Trip Time; the time it takes for a signal to travel from source to destination and back.
- RTX = Ray Tracing Texel eXtreme; NVIDIA's GPU series supporting real-time ray tracing and AI acceleration.
- TCP = Transmission Control Protocol; a reliable, connection-oriented transport protocol ensuring ordered and error-checked data delivery.
- UDP = User Datagram Protocol; a lightweight, connectionless transport protocol enabling low-latency communication without guaranteed reliability.
- VIP = Visually Impaired Person; an individual with significant vision loss for whom the navigation system is designed.
- WebRTC = Web Real-Time Communication; a peer-to-peer protocol optimized for real-time audio and video transmission.
- WebSocket = A low-latency, full-duplex communication protocol over TCP used to exchange video frames and metadata.
- Wi-Fi = Wireless Fidelity; a wireless networking technology based on IEEE 802.11 standards.

1 Introduction

1.1 Context

VIPs often use aids such as a white cane or more advanced systems like the I-CANE Mobilo [16]. However, as confirmed by a blind user, the I-CANE Mobilo cannot detect low objects or water. Such obstacles are only noticed when the tip of the (smart) cane accidentally strikes them. It does not provide any spatial mapping of the environment in front of the VIP.

There is a need for a more advanced detection system that leverages modern sensors and AI technology. Currently, no commercially available tools can perform detailed trail path planning in unknown environments such as parks.

Because the commercial incentive to develop solutions for these under-represented groups is too low, few trustworthy options are being created. This leads to a less inclusive society, especially when it comes to leisure activities. Simple things like making a walk in a park are nearly impossible for VIPs.

Advanced newer Edge-AI and three-dimensional (3D) vision-based systems like [1].lumen are promising solutions, but they require too much computational power to run in real-time on lightweight mobile platforms and if available, they are very expensive.

After conducting extensive research, as of today (6/10/2025), to our knowledge, there is no commercially available vision-based guidance system capable of guiding a blind person safely through a park with real-time trail path following.

Another important challenge is the difficulty of involving blind individuals in intensive testing. Organizing these field tests safely requires substantial support and guidance from sighted assistants, making the process complex and resource-intensive.



Figure 1: Blind person walking on custom trail (source: OpenAI DALL-E, 2025)

1.2 Objectives and contributions

1.2.1 Analyse Edge versus Cloud AI Performance for Real-Time blind Assistive trail path Navigation

The primary objective of this thesis is to evaluate the feasibility and performance of a real-time, vision-based assistive navigation system for VIPs that leverages mobile 5G connectivity and edge/cloud computing. Specifically, this work analyzes whether such a system can deliver sufficiently low latency, adequate frame rates, and reliable operation to support effective path planning and obstacle avoidance in dynamic outdoor environments, while maintaining acceptable power consumption and wearability.

A central research question is to what extent segmentation models can be executed on low-power mobile platforms while still achieving acceptable inference speed, frame rate, and energy efficiency. It is assumed that current affordable edge devices will not meet the requirements for real-time edge processing within a few milliseconds, particularly for segmentation tasks.

A secondary objective is to assess the impact of network conditions, such as variable latency and throughput, on system responsiveness and to evaluate strategies for maintaining safe operation under degraded connectivity in real-world scenarios.

1.2.2 Define methodology for Safe and Repeatable Evaluation

To achieve statistically robust performance measurements, the research will use both simulated environments and a robotic platform that emulates a blind user. This approach enables controlled experimentation and repeatability without exposing individuals to unnecessary risk. It is also important to consider security and privacy issues. Since a blind person cannot see what visual data is being transmitted, it is essential to address not only the technical aspects but also the ethical implications. Therefore, this thesis will pay special attention to the importance of end-to-end encryption.

1.2.3 Design proof of concept: 5G Trail path navigation for VIPs

The system architecture under investigation consists of a mobile end device that captures video data and transmits it over public 5G networks to an external GPU inference server. The server performs real-time image segmentation to identify traversable paths, returning guidance instructions (e.g., heading direction) to the robot.

Please note that close objects that appear suddenly outside the view angle of the camera while walking may be ignored. In this thesis, such near-field obstacles are assumed to be identified by complementary sensors, such as the (smart) cane, which serves as a local failsafe mechanism to ensure user safety. (Figure 2)

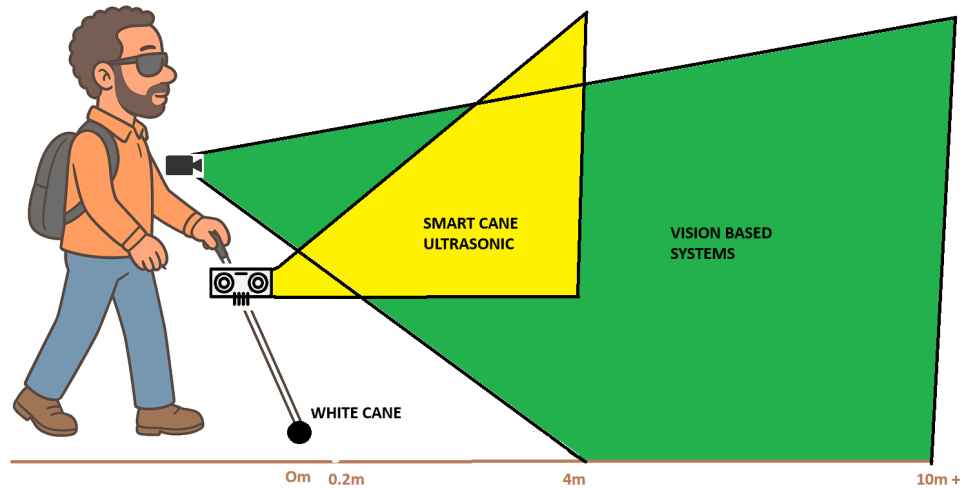


Figure 2: FOV of Smart Canes vs. Vision-based systems

1.3 Thesis Structure

This thesis is structured into 10 chapters.

Chapter 1: Introduction introduces the research context and motivation. It outlines the challenges faced by VIPs in outdoor mobility, defines the problem statement, and formulates the main objectives and contributions of the study, focusing on real-time path segmentation and safe navigation.

Chapter 2: Related Work reviews prior literature on assistive navigation, edge and cloud AI inference, and 5G-based robotics. It identifies the research gap that this thesis addresses, namely the absence of systematic benchmarking of segmentation-based navigation under real public 5G conditions with safety-oriented fallback mechanisms.

Chapter 3: Preliminaries provides background on positioning, networking, perception models, feedback systems, and security. Establishing the foundation for the experimental setup.

Chapter 4: Thesis Architecture Set-up details first the network setup, including remote GPU inference server, signaling server en Netgear M6 router and secondly the end-devices components, including the RPI4/5 edge devices, laptop and smartphone configurations.

Chapter 5: Methods presents the experimental design, including the creation of a digital twin in Unreal Engine, the FPV (first-person view) data generation, and the training of a lightweight segmentation model. It defines measurement procedures and additional performance metrics such as latency, jitter, queue size, and power consumption.

2 Related Work

Although progress has been made in both assistive navigation frameworks for VIPS and lightweight segmentation models for embedded platforms, there is very limited research that combines these domains under real network conditions. Few studies address the interplay between vision-based segmentation and real-time navigation for VIPs while explicitly benchmarking latency, variable bandwidth, jitter, and robustness to connection loss. In particular, systematic comparisons of edge versus cloud inference, including link stability and user safety through local failsafe mechanisms, remain largely absent. This gap motivates the present thesis, which integrates these dimensions into a unified 5G-enabled evaluation framework.

2.1 Edge inference

A recent contribution is the work of Ghahremanians and Mohammadi, who proposed a specialized semantic segmentation framework tailored to real-time pathfinding for VIPS [14]. Their two-branch architecture, combining a high-resolution “demarcation path” with a deeper “rich feature path,” achieves a strong balance between accuracy and speed, reaching 72.6% mean Intersection over Union (mIoU) on Cityscapes while sustaining 120 frames per second (FPS) on an NVIDIA Ray Tracing Texel eXtreme (RTX) 4090 graphics processing unit (GPU). This surpasses real-time baselines such as ESPNet and Fast-SCNN, and the authors further demonstrate robustness under adverse conditions (low light, weather, occlusion).

While this advances segmentation accuracy, it is tightly coupled to high-end GPUs, limiting its deployability. In contrast, the present thesis explores extremely lightweight segmentation models trained on Roboflow that remain feasible on lower-end hardware. By tailoring datasets to specific park environments, the models retain high quality without the prohibitive resource demands of state-of-the-art benchmarks. This complements architectural innovations by demonstrating an alternative path to practical, accessible deployment in 5G-enabled navigation.

Another smartphone-based initiative is the AI Guide Dog system [5]. Its design combines GPS and Google Maps with multimodal sensing (camera and IMU) to output short-horizon navigation cues through a classification model. Running locally on an iPhone 13 via CoreML, it sustains only 2 Hz guidance updates. The on-device approach strengthens privacy by keeping all data local but requires large datasets and a complex training pipeline. Compared to the present thesis, which emphasizes segmentation and real-world network variability, AI Guide Dog illustrates the trade-off between privacy-preserving inference and maintaining responsiveness on consumer hardware.

Other related studies, such as [19, 23], apply YOLOvX-based object detection to urban navigation. These approaches perform well in structured city environments, where abundant objects provide cues. However, object detection alone is less effective in unstructured or sparsely annotated environments such as parks or trails, where the critical task is continuous path segmentation rather than classifying discrete objects.

A noteworthy European initiative is the [1].lumen project, which develops AI-powered glasses that replicate the functions of a guide dog by interpreting the user’s surroundings and providing spatial feedback through audio and haptic cues [1]. Supported by the European Innovation Council, the project aims to combine wearable hardware and onboard AI to enable autonomous navigation for blind users without reliance on external connectivity. Unlike traditional smart canes such as the I-CANE Mobilo, which focus on obstacle detection near ground level, the *.lumen* glasses integrate real-time computer vision and scene understanding directly on the device. However, this approach requires significant local processing power and energy capacity, which constrains battery life and device weight. In contrast, the present thesis investigates offloading perception tasks to external GPU servers via 5G, thereby reducing the computational and thermal load on wearable devices while maintaining real-time responsiveness through optimized edge–cloud communication.

2.2 Cloud inference

One of the first systematic investigations into edge-assisted visual inference on mobile devices is the work of Gao et al. [12]. At ACM MobiCom 2019, they showed that offloading object detection to an edge server reduced latency to 60–100 ms over Wi-Fi and 150–200 ms over LTE, compared to over 500 ms for local inference on smartphones. Their study focused on object detection and did not cover segmentation, which remains more computationally demanding. The present thesis builds on this latency-focused work by extending it to segmentation under 5G networks and introducing additional metrics such as jitter, queue growth, frame-level reliability, power use, and the effects of compression/encryption overhead.

Azzino et al. present VIS4ION, a wearable platform that uses 5G edge offloading to support multiple AI microservices, including YOLOv7-based object detection and vision-language models [2]. Their adaptive streaming scheme (dual-stream encoding with REBERA link estimation) halved network latency on an OAI 5G testbed. While this demonstrates the potential of adaptive video pipelines, their evaluation remains confined to private

5G networks and focuses primarily on detection and language tasks. In contrast, the present thesis investigates segmentation under uncontrolled public 4G/5G network conditions, where variability and packet loss are critical factors for safe navigation.

Other publications such as [4] report latencies of 50 to 80 ms on direct links and 100 to 150 ms on public 5G, but only for raw video transmission without inference. Adding segmentation would inevitably increase delay, highlighting the need for integrated benchmarks that account for both networking and computation.

2.3 Overview

The present thesis distinguishes itself by systematically benchmarking edge versus cloud segmentation with a robotic simulator that allows safe and repeatable testing. Beyond accuracy, the evaluation targets end-to-end latency, jitter, and link stability under real public networks, while incorporating local failsafe mechanisms to maintain user safety under degraded connectivity. This explicit coupling of segmentation with networking variability addresses a gap that remains underexplored in prior work. An overview of the related work as discussed above can be seen in Table 1.

Table 1: Comparison of selected related works (2019–2025) by inference type and methodology.

Work	Focus	Inference Type	Methodology	Limitations	Relevance
[12] Gao et al. (2019)	Edge-assisted object detection	Network (Edge)	Offloading frames to GPU edge server over Wi-Fi/LTE	Only object detection; no segmentation; pre-5G	Early latency study; foundation for later 5G analysis
[2] Azzino et al. (2023)	Wearable AI over 5G	Network (5G Edge)	VIS4ION platform with adaptive streaming, YOLOv7 detection, and vision-language models	Controlled testbed only; no segmentation	Shows adaptive 5G streaming; complements focus on segmentation under real networks
[14] Ghahremanians & Mohammadi (2025)	Real-time segmentation for VIP pathfinding	Local High-End GPU	Two-branch network; 120 FPS on RTX 4090	High-end GPU dependence	Advances segmentation accuracy in contrasts with lightweight deployment focus
[5] AI Guide Dog (2025)	Smartphone-based navigation	Local (On-device)	GPS + Maps + Camera/IMU; classification at 2 Hz on iPhone 13	Limited update rate; no segmentation	Illustrates on-device privacy-preserving inference
[19, 23] YOLOvX-based methods (2025)	Object detection for urban navigation	Local(On device)	YOLOvX applied to dense city scenes	Less effective in unstructured trails; no path segmentation	Strong in cities; limited relevance for parks/trails
This thesis	Safe assistive navigation benchmarking	Edge–Cloud via 4G/5G	5G-enabled simulator; edge vs. cloud segmentation; real public networks	Simulator-based; segmentation limited to path/trail	First to couple segmentation with 5G variability and safety failsafes

3 Preliminaries

This chapter outlines the fundamental concepts and technical background required to understand the methodology and results presented in this thesis. It does not introduce novel contributions but provides definitions and context for the main building blocks of the system under study. Topics include positioning and mapping technologies relevant to outdoor navigation, the characteristics of contemporary 4G and 5G communication networks, and the role of perception models for path detection. In addition, mechanisms for user feedback are reviewed, together with security considerations and the use of simulation environments for safe and repeatable evaluation. These preliminaries serve to establish a common basis for the experimental design and subsequent analysis.

3.1 Positioning and Mapping

As reviewed in [26], several approaches have been proposed for path selection and navigation support for VIPs. However, GPS-based solutions that rely on open-source, crowd-sourced maps such as OpenStreetMap generally lack the granularity required to support reliable trail navigation. Recent research [30] demonstrates that dual-frequency Global Navigation Satellite System (GNSS) (L1+L5, e.g., available in recent Android and iPhone devices) can achieve accuracies of 30 cm to 1 m under ideal conditions. While applications exist that process raw GNSS data to further enhance accuracy to 30–50 cm, such methods remain unstable and insufficiently reliable for consistent real-world deployment.

3.2 Networking and Communication

Cellular networks(4G and/or 5G)

While much of the literature, such as [21], assumes access to controlled private 5G networks with strict Quality of Service (QoS) enforcement and latencies often below 10 ms, real-world deployment of assistive robotics for VIPs will more likely depend on public 4G/5G cellular connections provided like Proximus, Telenet or Orange via personal hotspots. These connections typically exhibit latencies of 20 ms or higher, along with increased variability and reduced reliability.

Via the mobile app BeCover+ of Test-aankoop, users can test their connection and this data is publicly available on the website [36]. The latency in the region Asse/Ganshoren is between 32 ms and 37 ms for operator Orange (used in this thesis). nPerf published a full report [28] on mobile network performance results in Belgium by different operators, as shown in Table 2.

Table 2: nPerf.com - Mobile network performance results (Belgium, 2024)

Provider	Downstream (Mb/s)	Upstream (Mb/s)	Latency (ms)
Proximus/Scarlet	105.4	17.0	29.8
Orange	96.7	18.1	32.1
Telenet	83.8	16.2	42.7
VOO	44.0	12.3	38.0

Exact analyses are not available in peer-reviewed literature, especially not for our specific test region (Zellik). As discussed in section 6.3, additional measurements (Table 6) were necessary to confirm the latency of the Orange 4G/5G connection. To contextualize these measurements, the Belgian Institute for Postal Services and Telecommunications (BIPT) provides reference data on network roll-out and quality. According to BIPT, as of March 2024, 5G coverage in Belgium reached approximately 87% of households outdoors and 68% indoors, complementing saturated 4G networks and improving overall latency and throughput [3]. Based on this, a wide range of latency values is included in Table 3, reflecting the variability across technologies and spectrum bands available in Belgium [35].

Table 3: Typical latency per technology in Belgium (RTT in ms)

Technology	Band(s)	Typical latency (ms)
4G LTE	B3 (1800), B20 (800), B7 (2600), ...	30–50
4G+ (LTE Advanced)	Carrier Aggregation	25–40
5G DSS / low-band	n1 (2100), n3 (1800), n28 (700)	20–35
5G n78 (3.5 GHz)	Sub-6 GHz high band	10–20
5G SA (standalone)	Dedicated 5G core	<10 (7–15 in practice)

As discussed in [22], the QoS and ultra-low latency requirements in robotic communication systems are critical, particularly when offloading high-dimensional sensor data (e.g., video) to edge servers, as any instability in the 5G link can compromise the safety and real-time responsiveness essential for navigation tasks in assistive robotics for the VIPs.

As mentioned by Liu et al. [22], the 5G radio signal might not be stable. Inspired by Simultaneous Localization and Radio Mapping (SLARM), new criteria must be designed [13]. One of the extra objective is to explore which criteria framework can support safe and reliable navigation for a mobile robotic platform.

With the continuous measurements of network quality (e.g., latency, framerate, frame loss, segmentation quality), the mobile device needs to remain aware of these criteria so that local failsafe systems can be activated when needed.

The SLARM-based mapping can serve as a decision-support layer for determining the optimal track and managing fallback or failsafe mechanisms, thereby enhancing both the robustness and intelligence of the assistive path planning system.

Current 4G-5G Coverage nPerf.com provides an overview of 4G and 5G coverage for different mobile providers. A snapshot of the region of interest is shown in Figure 3 and can also be accessed online [29].

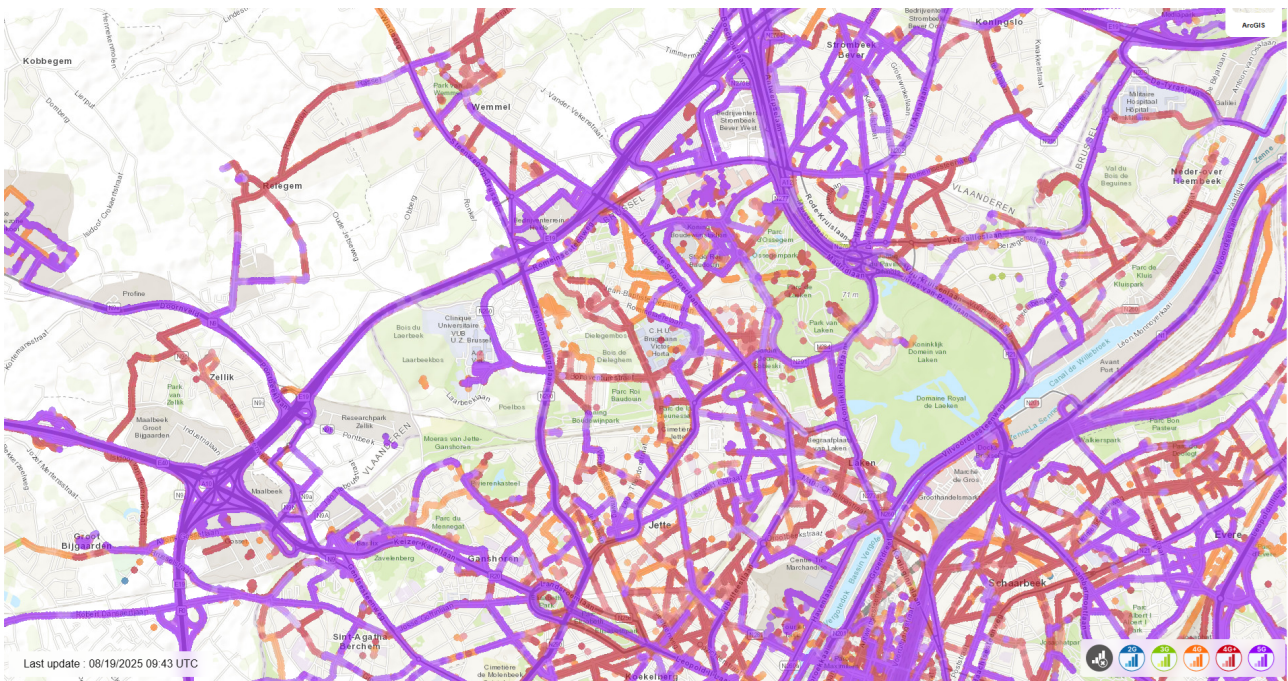


Figure 3: nPerf.com coverage of 2G–5G in the region of Zellik

The map indicates that 5G coverage is predominantly concentrated along major roads, university campuses, and densely populated areas. Therefore, it can be reasonably assumed that, in the current state of deployment, only 4G connectivity is available in parks and rural environments. However, as network deployment progresses, it is expected that providers will gradually extend 5G coverage to less densely populated areas, including parks and the countryside. This will ensure better performance in the future.

Impact latency and FPS on Real-time experience

As defined in [6], latency is defined as the total end-to-end time required for a video frame to travel from the camera (sensor) through encoding, wireless transmission, retransmissions, and network processing, until it is successfully received at the remote user or server. Low latency is essential to enable real-time responsiveness in systems where visual input is used for autonomous navigation or control. The shorter the delay between video capture, processing and feedback the more accurately and safely the robot can react to its environment. As described in [34], which discusses drones flying at much higher speeds than walking, latency is the most critical factor in real-time experience. An acceptable latency for FPV applications or systems were camera capturing and processing is involved, always depends on the movement speed. Walking (1.2-1.4 m/s) permits higher latency thresholds than fast aerial navigation (7 to 16 m/s).

In the context of this thesis, following latency threshold are taken into account:

- < 100 ms = ideal for walking speed
- 100–200 ms = good for walking speed
- > 200 –300 ms = noticeable delay

True end-to-end Latency As stated in [17], True end-to-end latency is defined as the time difference between capturing a frame by the camera (edge-device) and receiving the feedback on the edge-device from the processing server (GPU). In other words. The True end-to-end latency is defined as the total time between the moment you “observe” (capture) the world and the moment your system responds (e.g., displaying, acting, or receiving feedback). It points out that to calculate the True end-to-end latency accurately, it is essential to avoid any time synchronization issues, they relied on NTP-synchronized clocks. In our context it is difficult to use NTP. Because our end-device is the same as the haptic actuator, it is even more accurate to use the internal clock of the end-device.

As investigated in the [33] on low-latency, lossless link-layer design in data center networks, the Frame ID plays a central role, it is embedded in every data frame, tied to error checking, and governs retransmission.

To accomplish this logic in our setup, each frame carries a unique ID that is transmitted together with the frame content. The frame ID and corresponding timestamp are stored in a lookup table on the edge-device. Once the frame is processed by the server, the results are returned with the same frame ID. The edge-device can then calculate the latency by comparing timestamps for that specific frame using its own clock. This method does not require clock synchronization and enables precise calculation of the True end-to-end latency.

Calculate Latency when using external video inference server The calculation of the True end-to-end latency can be extended to include all relevant components of the transmission and inference pipeline. Similar decompositions of total latency, combining network transmission delay and inference (computation) delay, have also been used in recent work [24], where the end-to-end latency of collaborative inference is modeled as the sum of network and computation times. The extended formulation used in this thesis is expressed as:

$$\boxed{L_{\text{externalInference}}} = T_{\text{processing}} + T_{\text{transmission}} + T_{\text{propagation}} + T_{\text{inference}} + T_{\text{queue}}$$

$L_{\text{externalInference}}$ True end-to-end latency in experimental setup when using external inference server.

$T_{\text{processing}}$ This is the processing time. Compression time and encryption time were measured in the experiments on the edge-devices as they have a higher impact on low-end devices like Raspberry Pi’s (RPI). On the GPU server, there is no need to measure those.

$T_{\text{transmission}}$ Depending on the available transmission rate for the networks that are used, there can be a transmission latency of typically 20–35 ms in our region because we make use of domestic 5G networks. Also Network Address Translation(NAT), routing and other services causes additional delays.

$T_{\text{propagation}}$ This term represents the propagation delay, which depends on the physical distance between the sender and receiver and the speed at which signals travel through the medium ($\pm 3 \times 10^8$ m/s). Although propagation delay is typically negligible over short distances, it can become significant when the sender and receiver are separated by large geographic distances.

$T_{\text{inference}}$ This is the real inference time on the GPU server. The difference between the time just before inferencing started and right after it finished.

T_{queue} When the inference server receives new frames before it can process previous ones, the queue delay will rise. Therefore, it is important to monitor how many frames are waiting to be inferenced.

Calculate latency when inferencing on the end-device We are also experimenting with inference on the edge-device (section 4.1.2). In this scenario there will be no network transmission time or propagation time and no queue delay. Therefore the True end-to-end latency will be equal to the inference time and the processing time of the frame.

$$\boxed{L_{\text{localinference}}} = T_{\text{processing}} + T_{\text{inference}}$$

$L_{\text{localinference}}$ True end-to-end Latency in experimental setup when doing inference on the end-device.

$T_{\text{processing}}$ This is the processing time while the whole inference script is running. It includes capturing the frame and processing the feedback.

$T_{\text{inference}}$ This is the actual inference time, which is the difference between the time just before inference started and the time right after it finished.

Frames Per Second(FPS) In assistive navigation systems, the update frequency of feedback is critical for providing visually impaired users with timely guidance. While modern camera sensors are capable of sampling up to 60 Hz or more, Katzschmann et al. [18] demonstrated that haptic feedback does not need to match such high rate.

Depending on the situation, different FPS requirements apply. This finding confirms that haptic feedback in the order of 10 Hz is adequate for real-time guidance, even though the sensing system operates at much higher frequencies [18]. Higher frame rates in this context only increase CPU load without adding practical benefit.

If the recording FPS is set too high relative to the end-to-end latency in external inferencing mode, frame queues will start to fill up because frames are still being processed when new frames are available. This results in a significant increase in end-to-end latency.

When doing local end-device inferencing, there is no requested FPS as each frame will be processed as soon it is captured if the previous frame is processed.

Choice of WebSocket Communication over WebRTC

For real-time communication between the edge-device and the inference GPU server, this project uses WebSocket connections rather than WebRTC. WebSocket provides a full-duplex, low-latency Transmission Control Protocol (TCP) communication channel that is simple to implement and well-supported in Python. The protocol allows efficient transmission of JSON [9] metadata and binary-encoded frame data within a single connection, reducing overhead and complexity. Connection drops are automatically managed in the implementation, ensuring that disconnected clients are gracefully removed from the system.

In contrast, WebRTC[9] is primarily designed for peer-to-peer audio and video streaming in browsers and involves additional components such as ICE [9] negotiation, NAT traversal, and more complex signaling mechanisms. While WebRTC offers built-in media handling and UDP[9] transport for ultra-low latency, these features were not strictly necessary in this context, where the main requirement is reliable and ordered delivery of data over a stable mobile 5G link. WebSocket therefore presents a more lightweight and controllable solution better suited to this research topic.

These conclusions were made after some tests of different scenarios using both WebSocket and WebRTC implementations. A quantitative comparison is out of scope. All related source code is publicly available on a Github repository [8].

Domestic 5G networks as communication channel

The decision to use domestic 5G networks as the primary communication channel in this research is motivated by several practical and technical considerations. First, domestic 5G connections provided by commercial mobile operators (e.g., Proximus, Orange, Telenet) represent the most realistic scenario for real-world deployment. Unlike dedicated private (sliced) 5G networks, which offer guaranteed low-latency and high-throughput performance but require specialized infrastructure, public 5G services are widely accessible to consumers and can be used with standard mobile devices or portable routers.

To ensure stable and reproducible results and to obtain statistically valid measurements across multiple benchmarks and trials, a dedicated **Netgear Nighthawk M6 5G router** equipped with a mobile data SIM card was used during all experiments. This setup provides consistent 5G connectivity and minimizes variability introduced by changing network conditions, ensuring that performance differences can be attributed to the tested parameters rather than the network itself. In real-world deployments, however, such dedicated equipment is not required: A standard 5G-capable smartphone configured as a mobile hotspot can provide a comparable connection quality in practice, enabling practical and cost-effective field implementations of the system.

Secondly, the current 5G technology offers significantly higher bandwidth and lower latency compared to earlier generations (4G or 3G) of mobile technology, enabling now the transmission of compressed video frames in near real-time. Although latency and reliability are less predictable than in dedicated private (sliced) 5G environments, evaluating system performance under these conditions provides valuable insights into robustness and feasibility. Finally, using public 5G makes the solution inherently more scalable and economically viable, reducing barriers to adoption for assistive technologies targeting VIPs in everyday contexts.

3.3 Perception and AI Models

Instance Segmentation (Fast) COCO-compatible models

For path detection, this thesis employs a Roboflow 3.0 *Instance Segmentation (Fast)* model, initialized from a COCO-compatible checkpoint (COCO_n-seg). The COCO dataset is a widely adopted benchmark in computer vision, providing diverse annotations for object detection, instance segmentation, and keypoint estimation. Leveraging a COCO-compatible architecture ensures that the model benefits from standardized label formats, pretrained weights, and robust transfer learning capabilities.

The "Fast" variant prioritizes inference speed over maximum accuracy, making it suitable for real-time applications on resource-constrained edge-devices such as RPI boards or low-power laptops. Unlike heavier segmentation backbones (e.g., Mask R-CNN or DeepLabV3+), the lightweight design of the Roboflow Fast segmentation model trades off some detail in mask resolution for a significantly reduced latency and memory footprint.

In practice, this configuration achieves high precision on the limited path-versus-non-path task while remaining efficient enough to run in near real time. By training on a custom dataset of annotated park environments, the model adapts the general COCO features to the specific requirements of assistive navigation for VIPs. This approach highlights the feasibility of deploying segmentation-based guidance systems without relying on bulky GPUs, aligning with the thesis objective of enabling affordable and mobile solutions for outdoor navigation.

Row-wise midpoint extraction

In contrast to optimisation-based methods such as genetic algorithms [32], the approach in this thesis employs a row-wise midpoint extraction technique to generate feasible navigation paths. The principle behind this method is to divide the environment into a grid representation and evaluate each row independently. Within every row, the midpoint of the free space between obstacles is identified. By sequentially linking these midpoints across rows, a continuous path is formed that guides the VIP from its start position toward the target.

The resulting sequence of midpoints is averaged to compute a target heading direction. The heading angle is determined using a scanline-based centerline estimation algorithm. For each of several horizontal scanlines across the segmentation mask, the mean x-coordinate of the traversable area is computed. The aggregate target point is defined by the average x-position and the minimum y-position among all midpoints. The heading vector from the image's bottom center to this target point is calculated, and the angle relative to the horizontal axis is derived using the arctangent function.

This method offers several advantages. First, it ensures that each horizontal cross-section of the environment contributes only one candidate position, thereby reducing the search complexity compared to full path encodings that attempt to optimize over all possible trajectories. Second, because the midpoints maximize clearance from obstacles, the extracted path inherently improves safety margins without requiring additional collision-avoidance constraints. Third, the algorithm is computationally efficient: instead of evolving a population of candidate paths, it incrementally constructs a deterministic solution in real time.

The row-wise midpoint approach shares conceptual similarities with row-wise and column-wise path encodings described in Sedighi et al.'s work [32], where paths intersect each grid line exactly once. However, while Sedighi et al. relied on evolutionary optimization to determine feasible path vertices, our approach directly exploits the geometric structure of the free space. This makes it particularly suited for environments where rapid replanning is necessary, such as assistive navigation for VIPs. An example on how it was implemented can be seen on Figure 4.

3.4 Feedback and Interaction

Haptic feedback via arrow or servo

Findings from Katzschmann et al. demonstrate that VIPs can successfully interpret discretized haptic signals delivered at relatively low update frequencies (ex. 9 Hz) without perceiving delays, provided the mapping between sensor input and feedback is clear and consistent [18]. Translating this to an arrow/servo concept suggests that continuous angular output does not need to be updated at video frame rates, but rather at human-perceivable haptic update rates, ensuring responsiveness while avoiding unnecessary complexity or cognitive overload.

Importantly, this flexibility directly addresses several critical constraints identified in the interview, such as the limited usability of multi-cell braille displays during ambulation and the potential for sensory overload associated with glove-based haptic actuators. By abstracting the navigation command to a standardized directional output, the system architecture remains decoupled from any specific feedback hardware, which enhances extensibility and simplifies the integration of future assistive technologies.



Figure 4: Visual feedback for segmenting one example frame.

A further advantage of this approach lies in its high degree of modality independence. Because the directional output is inherently quantitative (i.e., a numeric heading or relative angle), it can be systematically transformed into alternative feedback channels, including haptic or auditory modalities. For instance, the angle of the arrow can be mapped to vibration intensity, frequency, or spatial haptic patterns delivered via actuators.

Confirmed by the interview with a VIP, the adoption of an *arrow-based feedback system*, wherein directional guidance is encoded as a visual arrow indicating the recommended heading, offers several distinct advantages from both a usability and a system design perspective. Firstly, directional arrows constitute an intuitive and universally recognizable representation of orientation. This encoding facilitates rapid interpretation with minimal cognitive load and does not require extensive training or prior familiarity. Moreover, unlike text or braille-based instructions, arrows are language-independent, which enhances accessibility. An servo-based haptic braille feedback prototype, available via Tinkercad [9], was developed during this research and evaluated with a visually impaired participant (VIP), as shown in Figure 5.

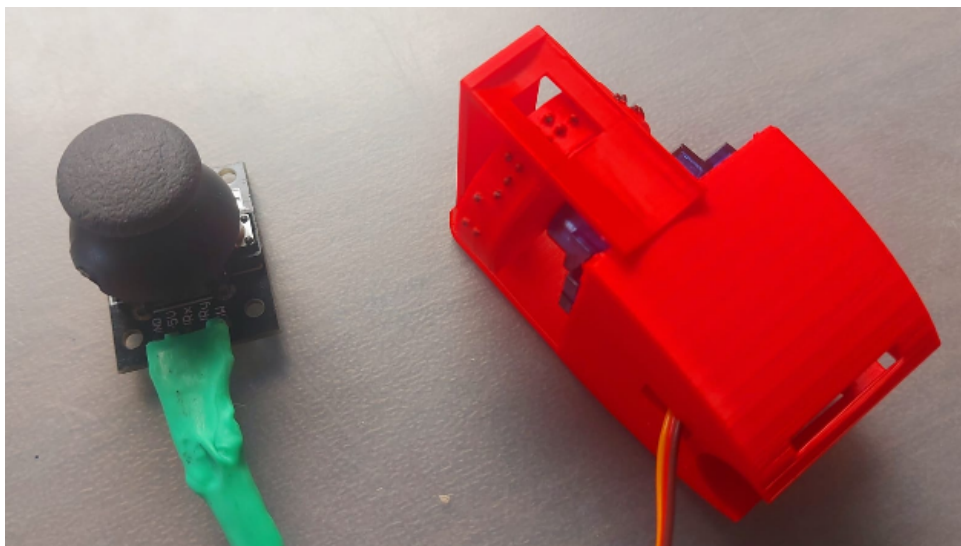


Figure 5: Haptic braille feedback system via Servo

3.5 End-to-end Encryption via AES-256

To secure the video data, AES-256 encryption in Cipher Block Chaining(CBC) mode with PKCS#7[9] padding is applied. This encryption method is considered industry-grade and is recommended by the NIST as a standard for secure communications [25, 7].

A unique initialization vector (IV) is generated for each frame using `os.urandom(16)`, which prevents pattern formation in the encrypted data stream and ensures semantic security. Encryption occurs before transmission via WebSocket, thereby providing full end-to-end protection. Even the signaling server has no access to the visual content. It just forwards the encrypted frame to the GPU server.

It should be noted that CBC mode by itself does not provide integrity protection, and the use of a fresh IV does not establish forward secrecy. For production systems, an authenticated encryption scheme such as AES-GCM (or alternatively CBC combined with a message authentication code, e.g., HMAC-SHA256) should be used. Metadata such as timestamp, compression time, and resolution are currently transmitted in plaintext to facilitate local analysis.

In the prototype implementation used for this thesis, no key management mechanism was included in order to streamline development and improve benchmarking speed. However, for deployment in a production environment, a secure key management system must be implemented, as discussed in the Future Work section.

3.6 Unreal Engine 5.6 as a Realistic Simulation Environment

Unreal Engine is a state-of-the-art 3D creation platform originally developed for the video game industry, but increasingly adopted in robotics, autonomous driving, and virtual prototyping. Its ability to render photorealistic graphics, simulate complex environments, and provide real-time performance makes it highly suitable for developing digital twins of real-world outdoor settings. In this thesis, Unreal Engine 5.6 is employed to construct a controllable and safe testbed for evaluating vision-based navigation systems for VIPs. The platform allows rigorous experimentation without exposing participants to unnecessary risks, while maintaining compatibility with the same AI inference and networking pipelines used for real-world video data.

The main advantages/risks of using Unreal Engine in this context include:

- **Rapid prototyping and iteration:** Simulation-based testing accelerates development cycles by reducing reliance on physical prototypes and enabling quick validation of design choices.
- **Systematic stress testing:** Rare or hazardous conditions (e.g., sudden obstacles, occlusions, adverse lighting) can be created and repeated on demand to evaluate system robustness.
- **Integration with AI pipelines:** Rendered camera streams can be processed through the same compression, encryption, and inference chain as real-world data, ensuring realistic benchmarking.
- **Cost efficiency:** By minimizing the need for repeated field trials and specialized hardware during early development, the simulation environment lowers resource requirements.
- **Risk of overfitting:** Simulation environments introduce a risk of overfitting, as models may learn simulation-specific visual or physical characteristics that do not fully to real-world conditions.

4 Thesis Architecture set-up

After the preliminaries, we can now begin explaining the thesis. First, we describe the architecture, including network architecture and End-Devices, then the methods used. Lastly the results of this approach are shown using several benchmarks.

4.1 Network architecture

Following schema, Figure 6 has been used for the setup. You can see two different paths or testing scenarios. One route is used to do the benchmarks via domestic 5G mobile networks(RPI, see section 6.1.1 and Laptop, see section 6.1.1) and the other route is used for the local Wi-Fi setup (RPI, see section 6.1.1 and Laptop, see section 6.1.1). The outbound route is shown in green arrows. The Return path is shown in orange arrows.

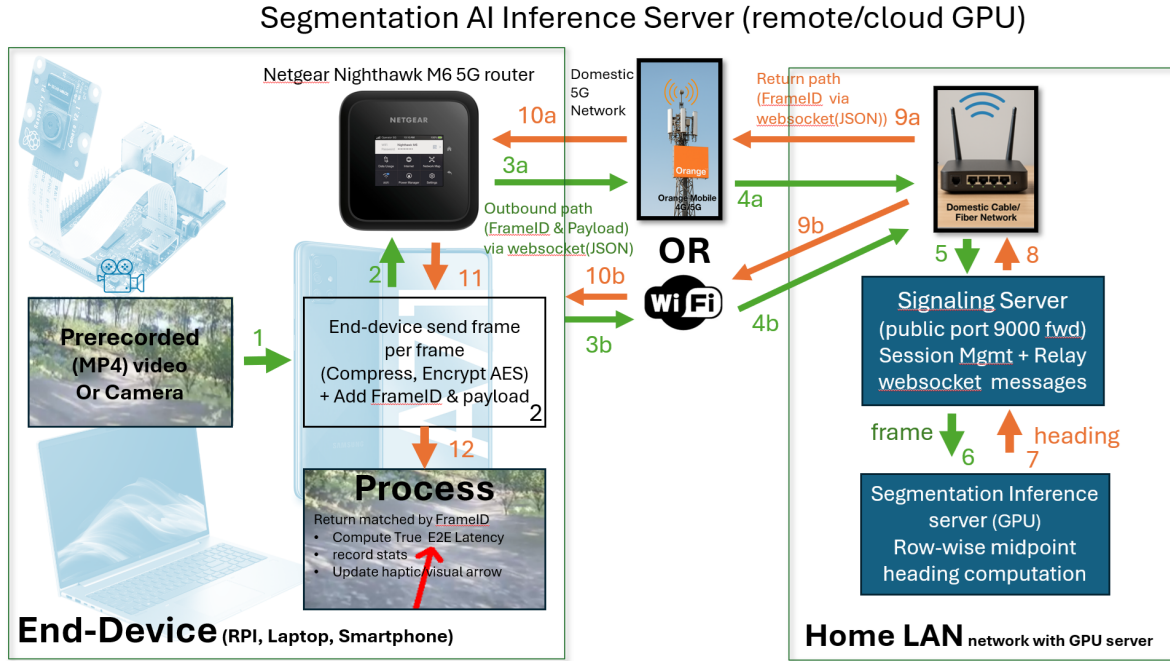


Figure 6: Benchmark setup for real-time performance testing (green: outbound path, orange: return path) with numbered arrows indicating the data flow.

4.1.1 Domestic 4/5G network route

When using the 4/5G network, all outbound traffic (FrameID, content and extra payload) is sent to the publicly available "Home LAN network with GPU server" via the Netgear 5G router. When the GPU computed the heading, the Return path (orange arrows) over the same 4/5G connection is used to send back the heading and inference time.

4.1.2 Local Wi-Fi network route

When using the local Wi-Fi configuration, all components (end-device, signaling server, and GPU inference server) are connected within the same Wi-Fi network. Unlike the 5G setup, no mobile router or NAT traversal is required. The local router (mainly the Domestic Cable/Fiber modem) will manage all connections. The end-device sends the FrameID and payload to the signaling server (green Outbound path) and gets back the heading and inference time (Orange return path) over the same Wi-Fi network.

4.1.3 Segmentation AI Inference Server (remote/cloud GPU)

For the purpose of this thesis, a gaming desktop tower was employed as the primary server platform. This type of setup is common and relatively affordable, while still providing full and dedicated control over the machine during the research phase. Equipped with a consumer-grade GPU, the desktop offers sufficient computational power to run segmentation models in real time and to evaluate the end-to-end pipeline under realistic conditions.

Although high-end GPUs such as the recently released Nvidia RTX 5090 are now available, their cost remains prohibitive for many research and development projects. As demonstrated in later sections, the inference time on the GPU accounts for only a small fraction of the total end-to-end latency; transmission, compression, and encryption delays are often more dominant. This means that the use of a top-tier GPU provides limited benefits in the context of single-user, real-time assistive navigation. Nevertheless, a more powerful GPU could be advantageous when scaling up to support multiple inference tasks simultaneously, for example if additional users or extra functionalities were to be added in the future.

Following the research phase, it is recommended to migrate towards a cloud-based infrastructure or a rack-mounted server solution that offers built-in redundancy, backup facilities, and higher guarantees of uptime, while maintaining the flexibility to allocate additional computational resources as needed.

Specifications

- **Hardware model:** Alienware Aurora Ryzen Edition
- **Memory:** 32.0 GiB
- **Processor:** AMD Ryzen 9 3900X 12-core processor $\times$ 24
- **Graphics:** NVIDIA GeForce RTX 2080 Ti (TU102)
- **Disk capacity:** 2.0 TB
- **OS Name:** Ubuntu 22.04.5 LTS
- **OS Type:** 64-bit
- **GNOME Version:** 42.9
- **Windowing System:** X11

4.1.4 Local Signaling Server

It is implemented as a service (python script) on a Synology NAS-system within the same LAN of the GPU server, which guarantees full control over the signaling process and independence from third-party services.

The signaling server in this thesis serves as a lightweight coordination point between sender and receiver. It provides a persistent WebSocket channel over which both sides exchange metadata such as frame identifiers, encryption information, and connection status updates. Importantly, the signaling server does not transmit or process video streams: The actual image data is compressed and encrypted at the sender and then sent directly to the inference server. Its role is therefore limited to synchronizing communication and ensuring that both endpoints remain aligned during real-time experiments.

By enabling basic port forwarding (port 9000) on a standard domestic cable or fiber connection, the server can be made publicly accessible over the internet. This configuration allows the GPU inference server to be reached from any remote 5G connection without requiring a dedicated Virtual Private Network (VPN) or relay infrastructure.

While this setup is technically effective for research purposes, it should be noted that exposing services directly to the internet without additional safeguards is not the most secure approach.

4.1.5 Netgear Nighthawk M6 5G router

The Netgear Nighthawk M6 is a portable 5G router designed to provide reliable broadband connectivity in both stationary and mobile contexts. It is built around the Qualcomm Snapdragon X62 modem, which supports 5G Sub-6 GHz operation with peak downlink rates up to approximately 2.5 Gbps under optimal conditions [27]. The device integrates IEEE 802.11ax (Wi-Fi 6) functionality, enabling simultaneous connections for up to 32 client devices, while also offering a gigabit Ethernet interface for wired backhaul. From a usability perspective, the unit is equipped with a 2.4-inch control display and supports operation either via an adapter or via a rechargeable battery, providing up to 13 hours of autonomous use or continuous mains power. Experimental evaluations



Figure 7: Netgear Nighthawk M6

indicate that the device’s effective coverage is around 2000 m² in indoor environments when operated without battery constraints. The Nighthawk M6 also incorporates external antenna ports, allowing researchers to investigate the effect of directional antennas on throughput and signal stability. However, because the router is limited to Sub-6 GHz 5G frequencies, its performance is constrained compared to millimeter-wave (mmWave) 5G systems, particularly in high-capacity scenarios.

4.2 Edge-devices

4.2.1 Raspberry Pi 4 / 5

The RPI was selected as the edge-device in this study because it offers an affordable, compact, and sufficiently capable platform that reflects the practical constraints of deploying assistive AI systems for VIPs. Its full support for Python programming and a wide range of open-source libraries makes rapid prototyping straightforward, while General-Purpose Input/Output (GPIO) headers and Universal Serial Bus(USB) interfaces enable direct connection to cameras, sensors, and external modules such as GPS modules. When paired with a lightweight battery pack, the RPI can be carried unobtrusively in a small bag, allowing fully mobile operation in outdoor environments. The availability of pre-configured operating system images on microSD cards further simplifies deployment and ensures reproducibility across multiple devices.

In this thesis, the RPI primarily functions as a frame capture and transmission unit; It records camera images, compresses and encrypts them, and sends them to the inference GPU. Once results are returned, the RPI also delivers real-time feedback to the user via visual, audio, or haptic channels. For benchmarking, a RPI5 was used to evaluate performance over both 5G and Wi-Fi networks (see Section 6.1), while both RPI4 and 5 devices were tested for local on-device inferencing of segmentation models (see Section 6.2).

Specification	Raspberry Pi 4 Model B Rev 1.5	Raspberry Pi 5 Model B
Processor cores	4 (ARM Cortex-A72)	4 (ARM Cortex-A76)
Base frequency	1.5 GHz	2.4 GHz
BogoMIPS per core	108.00	153.60
CPU architecture	ARMv8-A (64-bit)	ARMv8-A (64-bit)
Supported instructions	fp, asimd, evtstrm, crc32, cpuid	fp, asimd, evtstrm, crc32, cpuid
CPU implementation	0x41 (ARM)	0x41 (ARM)
CPU model	0xd08 (Cortex-A72)	0xd0a (Cortex-A76)
RAM type	LPDDR4-3200	LPDDR4X-4267
PCIe support	No native PCIe	PCIe 2.0 x1 slot
Hardware	Broadcom BCM2711 (SoC)	Broadcom BCM2712 (SoC)
Board revision	b03115	c03140

Table 4: Comparison of RPI4 and RPI5 specifications

4.2.2 Standard laptop

In addition to edge devices such as the RPI, a standard laptop was used throughout this thesis to accelerate development, testing, and benchmarking. The laptop provides significantly more computational resources than lightweight edge hardware, making it particularly suitable for rapid code iteration, debugging, and running complex experiments that would be time-consuming or impractical on a RPI. By ensuring that the same scenarios and pipelines can be executed both on the laptop and on constrained devices, the development process benefits from faster prototyping cycles while maintaining consistency in evaluation.

The system used is an HP EliteBook 860 G10, configured as follows:

- **Description:** HP EliteBook 860 16-inch G10 Notebook PC
- **Installed RAM:** 32 GB
- **Processor:** 13th Gen Intel(R) Core(TM) i5-1335U
- **Operating system:** Windows 11 Pro v24H2
- **System type:** 64-bit operating system, x64-based processor

The laptop was primarily employed for code development, simulation experiments, and baseline benchmarks of the video streaming and inference pipeline under various network scenarios. This dual-device setup (laptop and RPI) enabled a balance between fast, flexible experimentation and realistic validation under constrained edge-computing conditions.

4.2.3 Smartphone (Stay on trails app)

As part of the master's course Transmission Techniques and Multimedia, the Android application Stay on Trails (Figure 25), introduced in Section 8.2, was developed as a selected topic to evaluate whether the core principles of real-time path segmentation and heading estimation can be implemented on widely available mobile devices. The application runs on a standard Android smartphone, such as a Samsung Galaxy A71, and integrates a lightweight version of the cloud-assisted navigation pipeline, including GPS logging and end-to-end latency measurements.

5 Methods

This study follows a reproducible pipeline that couples a controllable *digital twin* of an imaginary park, vision-based perception, and a communication/inference stack that mimics real-world deployment while remaining safe and repeatable.

5.1 Experimental setup

5.1.1 Digital twin and FPV data generation

A photorealistic park environment was built in Unreal Engine (UE 5.6), a view from above can be seen on Figure 9. A minimal *first-person view* (FPV) game actor (Figure 8), configured with human walking kinematics and virtual camera, simulates the user’s “eyes.” Camera motion (speed, head sway, turns) and environmental factors (lighting, obstacles) are scripted so that identical traversals can be replayed. The FPV camera renders (RGB)[9] frames that represent what a VIP would “see” through a wearable device.



Figure 8: Unreal Engine FPV



Figure 9: Unreal Engine 5.6 - Map of Digital twin of park

5.1.2 Segmentation model for path detection

From the FPV footage, representative frames were annotated to label the traversable path versus non-path classes. A lightweight, COCO-style instance segmentation model (Roboflow 3.0 “Instance Segmentation (Fast)”) was trained and exported for real-time inference. At runtime, the model produces a per-frame path mask, a

row-wise midpoint extraction converts this mask into a centerline and a target heading angle, which drives the haptic/visual arrow interface.

Training A limited dataset of 231 video frames were extracted from a prerecorded tour in the Unreal Engine “twin city” park environment and split into train/validation/test = 162(70%)/47(20%)/22(10%). Images were resized to 640×640 (average size ≈ 0.41 MP). Annotation was performed in Roboflow (see Figure 10).



Figure 10: Example of some annotated images

The Roboflow 3.0 Instance Segmentation (Fast) model, initialized from the `COCOn-seg` checkpoint, was fine-tuned on our custom dataset (`unrealsim/1`) containing COCO-style instance annotations. During preprocessing, images were automatically oriented and resized to 640×640 pixels, without applying any data augmentation. The model was trained for 320 epochs, and the results reported in this thesis correspond to the best-validation checkpoint obtained at epoch 119.

Validation performance The segmentation model is trained to predict the walkable path in a park environment. On the validation set, it achieves a mask precision of 0.998 and a recall of 1.00, indicating that almost all walkable regions are correctly detected while virtually no non-walkable areas are incorrectly classified as path. The mean average precision at IoU 0.5 (mAP50) for the walkable-path masks reaches 0.995, and the more stringent mAP50–95 metric still yields 0.877. This shows that the model not only detects the presence of the path but also delineates its boundaries with high spatial accuracy, which is crucial for deriving a reliable center line for navigation. These metrics suggest that, within the validation domain, the risk of guiding a user towards non-walkable regions (false positives) is very low, and the available path width is largely preserved (few false negatives). Nevertheless, real-world deployment in different parks, seasons, and lighting conditions may introduce domain shifts; therefore, these quantitative results are complemented with qualitative field tests and additional safety mechanisms.

Deployment The trained weights are stored in file `unrealsim.pt` (PyTorch format). During inference on the GPU server, the model is loaded from this file to perform real-time segmentation of camera frames and yield compact guidance messages including the unique FrameID.

```

=== Parameter stats ===
Total parameters:      3,263,811
Param memory (fp32):   12.45 MB
=== Computation cost ===
Input size:            1 x 3 x 640 x 640
MACs:                  6.05 GMac
Approx FLOPs:          12.11 GFLOPs (assuming 2 FLOPs/MAC)
=== Inference benchmark ===
Device:                cuda
Runs:                  50
Avg time per image:    5.64 ms
Throughput:            177.42 FPS

```

5.1.3 Reproducible streaming playback

To decouple perception from data collection and to enable fair comparisons across devices and networks, experiments use a *prerecorded video* of 9-minutes and 43-seconds MP4 video of a fixed park traversal. This recording resulted in 17490 frames. The same frame sequence is replayed in every run during benchmarks, see section 6.1, ensuring that variations in results stem from compute and network conditions rather than scene changes.

Streaming and inference pipeline. On the edge-device, each replayed frame is timestamped and assigned a **FrameID**, compressed and encrypted, and sent over a WebSocket connection via a 4G/5G link to a publicly reachable signaling server. The signaling server relays frames to the GPU inference server on the same LAN. The server runs the segmentation model, computes the heading for the corresponding **FrameID**, and returns a compact message containing **FrameID** and heading. via the same route. The edge-device (ex. RPI) matches responses by **FrameID**, updates the arrow/servo output, and logs performance metrics to a Comma-separated file (CSV) file. As a demonstration, the processing of one example frame is shown in Figure 11.



Figure 11: Realtime processing including performance measurements

Using services for stable inference To guarantee that the video segmentation service is executed automatically and remains reliable during operation, a dedicated **systemd** service is running on the Ubuntu 22.04 GPU. The service uses the Python interpreter from the project's virtual environment to launch the segmentation script `segmentVideoStream.py` with the selected model file. It starts only after network connectivity is established (`After=network.target`) and it automatically restarts on failure with a 5-second delay (`Restart=always, RestartSec=5`). This combination of **systemd**-based management and scheduled monitoring ensures fully unattended operation of the segmentation pipeline, while also providing automatic recovery in case of unexpected crashes or interruptions. This method is essential to assure good connectivity with the inference server when in the field with minimal failures.

5.2 Measurement design

Accurate and consistent measurement of network and system performance is essential to evaluate the feasibility of real-time assistive navigation. In this work, the true end-to-end latency per frame is computed on the edge-device using the internal clock in combination with the unique **FrameID**. This latency reflects the total time between capturing a frame and receiving the corresponding inference result (processing + transmission/propagation + queuing + inference). In parallel, the system records FPS, queue size, and error conditions. Because the same prerecorded video is replayed in every benchmark, repeated runs under different configurations (e.g., local Wi-Fi vs. 5G, edge vs. cloud inference) at different bitrate/FPS settings yield statistically comparable results. All metrics are stored in CSV format for later analysis. The complete list of logged fields is provided in Table 5.

5.2.1 Standard 5G metrics in related research

Existing studies on 4G/5G performance typically focus on a limited set of key indicators:

- **Latency** (one-way, round-trip, or end-to-end), often reported as the primary determinant of real-time responsiveness.
- **Jitter**, or the variance in latency over time, which indicates stability of the link.
- **Throughput** (uplink and downlink), reflecting the available data rate.
- **Packet loss**, representing the percentage of transmitted frames or packets that fail to arrive.

These metrics are widely adopted in the telecommunications and robotics literature, as they capture the fundamental QoS characteristics of wireless networks. The metrics that are used in this thesis are shown in Table 5.

5.2.2 Additional metrics introduced in this thesis

In addition to the commonly used standard set of metrics, this thesis introduces further metrics to capture the interaction between networking, computation, and power consumption in an edge/cloud inference pipeline. These include:

- **Processing overhead** such as compression time per frame and encryption time, which can dominate latency on low-power devices.
- **Inference time**, measuring the duration of model execution on the target hardware.
- **Queue size**, representing the number of frames waiting for processing, which directly impacts responsiveness under load.
- **Power consumption**, to evaluate the feasibility of battery-powered deployment.
- **Frame-level reliability indicators** such as missed frames and successful frames, providing a fine-grained view of robustness.

By incorporating these additional parameters, the measurements go beyond connectivity evaluation and provide insight into the true end-to-end behaviour of the complete assistive navigation system.

5.2.3 Summary of logged fields

An overview of all recorded metrics, their definitions, and units is provided in Table 5.

The fields are grouped into categories for time and identification, network and connection, video quality and compression, and system performance and AI. This grouping clarifies the relationship between standard QoS metrics and the additional contributions, marked with (*), specific to this thesis.

5.3 Feedback visualization

5.3.1 Visualisation of haptic feedback

The inference service implements an asynchronous, WebSocket-based segmentation pipeline: Incoming frames are segmented on the GPU server, a heading angle θ is computed, and the result $\{\text{FrameID}, \theta\}$ is returned to the Edge-device to drive the haptic arrow/servo. During normal operation, no on-screen overlay is shown, it's only used for analysis and documentation.

As shown in Figure 4, A debug visualisation can be enabled that overlays the segmentation mask, row-wise midpoints/centreline, and the computed heading.

Table 5: Overview of measured fields, grouped by category, with descriptions and units

Time and Identification		
datetime	Timestamp when the measurement was recorded.	ISO 8601
last_frame_id	Identifier of the last processed frame.	ID
gps_coordinates	Geographic location of the device.	latitude, longitude
Network and Connection		
signaling_server	IP of the signaling server	LAN or WAN IP
Connection	Type of network connection used.	category(Wi-Fi, 4G, 5G)
bitrate	Data transmission rate.	Mbps
latency_ms	Per-frame end-to-end latency	ms
Video Quality and Compression		
resolution	Video resolution in pixels (width $\times$ height).	pixels
max_fps	Maximum negotiated frame rate.	fps
jpeg_quality	Compression quality setting for JPEG encoding.	10–100
current_fps	Effective frame rate achieved.	fps
current_size_kb	Compressed frame size.	kB
compression_ms *	Time to compress a single frame.	ms
encryption_ms *	Time to encrypt a single frame.	ms
System Performance and AI		
poweruse_W *	Power consumption.	W
inference_ms *	Time required for AI inference on a single frame.	ms
queue_size *	Number of frames waiting in the queue.	count
missed_frames *	Total frames dropped before processing or transmission.	count
successful_frames *	Total frames successfully transmitted and processed.	count

5.4 Benchmark setups

To address the first objective in Section 1.2.1, benchmarks were conducted using either a standard laptop or a RPI5 as end-device. The benchmark process pipeline can be seen on Figure 12.

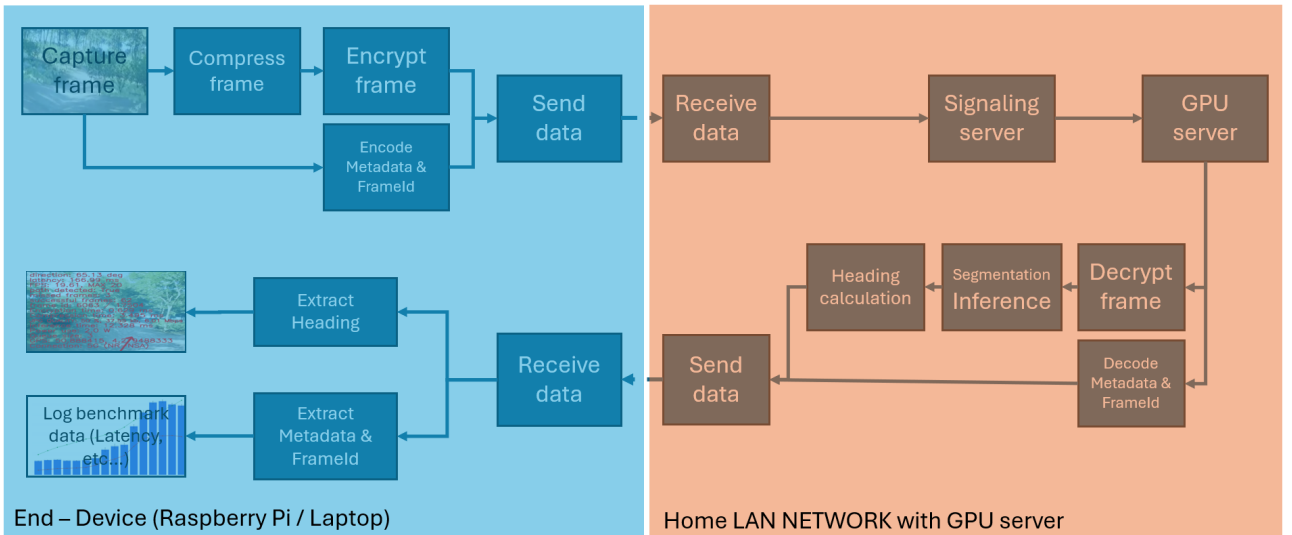


Figure 12: Benchmark pipeline process from capturing frame to visualise and log metric.

Assuring best performance As stated in section 3.2, 5G Standalone (SA) deployments in Belgium are limited to testbeds and are not yet widely rolled out. This thesis only covers 5G Non-Standalone(NSA) coverage. To ensure consistent and reliable results, benchmarks on the end-devices were conducted from a fixed location (Den Ham 6, BE-1731 Zellik), where 5G NR in NSA mode is available. (see Figure 3). With the use of the GPS coordinates, these measurements can then later be used to compare with the field latency stability tests as performed in Section 6.3.

The end-devices were responsible for reading frames from a prerecorded MP4 video, compressing and en-

crypting each frame, adding a unique **FrameID**, and transmitting the data over a 5G connection via WebSocket to the signaling server. The signaling server, hosted on a Synology NAS with public port 9000, relays the frames to the GPU inference server. On the GPU server, row-wise midpoint segmentation is performed to compute a heading direction. This heading direction and the inference time is then returned together with the corresponding **FrameID** via the same signaling server. The end-device then matches the response with its original FrameID, computes the True end-to-end Latency (see Section 3.2) and other metrics as in Table 5 and updates the haptic/visual arrow output.

6 Benchmark measurements

6.1 Real-time inference via an external GPU

During playback of the pre-recorded FPV sequence (Section 5.1.3), all per-frame metrics listed in Table 5 were logged to CSV as described in Section 5.4.

To evaluate real-time performance (Section 3.2), we increased the requested frame rate (`max_fps`) in steps of 2 FPS every 100 frames: 10, 12, ..., 40 FPS (16 settings). From a total of 10000 frames, this yields approximately 625 frames per FPS setting.

Latency experiments used JPEG quality 90% at 640×480. The median encoded frame size was ≈ 109 kB (i.e., ≈ 8.7 – 34.9 Mbit/s over 10–40 FPS). For each FPS setting we report the sample mean and a 95% confidence interval (CI) for latency computed as on per-frame values:

$$\bar{x} \pm t_{0.975, n-1} \frac{s}{\sqrt{n}},$$

where n is the number of frames in that setting, $\bar{x}$ is the sample mean, and s the sample standard deviation. Queue size and bitrate are reported as sample means per setting.

6.1.1 Real-time responsiveness benchmarks (Latency, FPS and bitrate)

RPI5 connected via local Wi-Fi network

The RPI5 benchmark over local Wi-Fi is shown in Figure 13. Results demonstrate stable throughput and limited queue growth up to 20 FPS. Starting from 30 FPS the latency increases faster. But still the 200 ms threshold is only reached at approx. 36 FPS. Beyond this point, latency increases due to buffer build-up: the GPU inference server cannot keep pace with the requested frame rate. With JPEG quality fixed at 90%, the RPI5 sustains real-time responsiveness for walking speeds (latency < 200 ms).

As expected, bitrate grows nearly linearly with the requested FPS. With an average frame size of ≈ 109 kB, the theoretical bitrate is 109 kB X FPS (e.g., 21.8 Mbit/s at 20 FPS). The measured bitrate is consistently lower due to processing delays.

Queue size and latency follow the same trend. Latency is composed of transmission and inference time: When frames accumulate, the additional delay scales approximately with $queue\ size \times inference\ time$. For example, with an inference time of 12 ms and a queue of 10 frames, latency increases by about 120 ms.

To keep latency within acceptable bounds, the next frame should not be sent before the previous one is processed. In our testbench, this was enforced by inserting a controlled delay after each frame to regulate the effective FPS.



Figure 13: Benchmark RPI5 Wi-Fi (JPEG quality 90%) real-time external video Inferencing (mean 95% CI)

RPI5 connected via domestic 5G mobile network

When connected via the Netgear M6 5G router (Figure 14), the RPI5 showed higher latency compared to the local Wi-Fi setup, with a gradual increase already visible from 18-20 FPS. On 5G, latency crossed the 200 ms threshold consistently at 26 FPS and rose steeply beyond 28 FPS, reaching over 400 ms at 32 FPS and approaching 500 ms at 40 FPS, with queue peaks approx. 10 frames at 34-38 FPS. This behavior is expected due to the additional uplink and downlink propagation delays and the inherent variability of public 5G networks.

Queue size followed a similar trend as it is relatively stable up to 20 FPS, but grows steadily afterwards and peaks around 10 frames between 34-38 FPS. Unlike the Wi-Fi case, where queues remained small until higher frame rates, the 5G connection showed more pronounced sensitivity to transient fluctuations, with occasional jitter and frame drops.

Overall, while short-term averages remained acceptable for walking speeds at lower frame rates (≤ 22 FPS), pushing the system above this envelope led to clear instability. To ensure safe real-time assistive navigation, the RPI5 on domestic 5G should therefore be operated conservatively below 24 FPS.

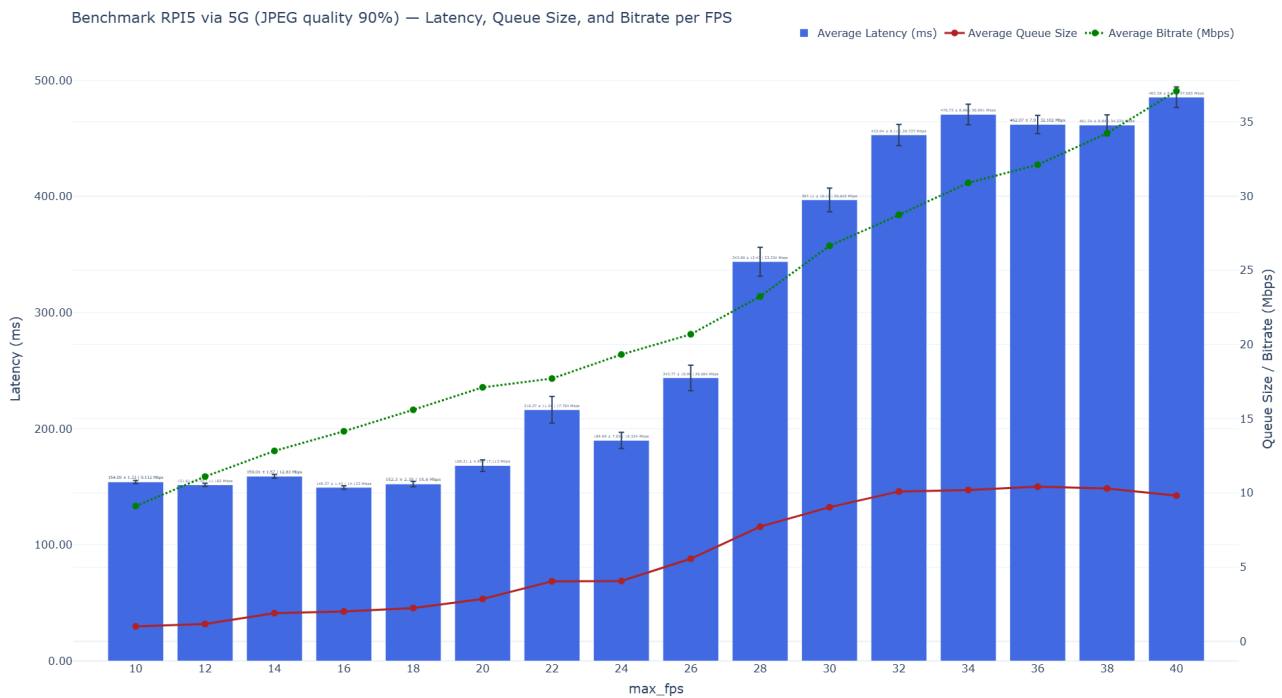


Figure 14: Benchmark RPI5 on 5G (JPEG quality 90%) real-time external video Inferencing (mean 95% CI)

Laptop connected via local Wi-Fi network

The benchmarks reveal that the RPI5 actually maintained more stable latency scaling over Wi-Fi than the laptop. On the RPI5, latency stayed below approximately 120 ms up to about 28 FPS, with only a gradual rise before sharp queue growth appeared beyond 34 FPS. By contrast, the laptop, despite its stronger CPU, pushed more frames into the pipeline, leading to earlier queue buildup and a noticeable latency increase already from 22-24 FPS. At 30-32 FPS, the laptop exceeded 120 ms and started rising faster, while the RPI5 only began rising more sharply from 36 FPS.

This shows that the RPI's lower compression throughput, while seemingly a disadvantage, actually acts as a form of natural flow control that keeps the GPU server's queues manageable. In other words, lower compression throughput on RPI5 limits ingress rate and prevents queue blow-ups, yielding more predictable latency than the laptop at the same requested FPS. The laptop can achieve higher bitrates, but this comes at the cost of pushing the server into overload earlier, undermining real-time stability. The trade-off is clear: the RPI5 offers more predictable responsiveness within the safe walking envelope of 28 FPS, while the laptop only performs better in raw throughput terms but suffers from earlier latency degradation.

Laptop connected via domestic 5G mobile network

When connected via the domestic 5G network (Figure 16), the laptop exhibited the same overall behavior as the RPI5, though performance degraded slightly earlier. Latency remained between 120-150 ms at 10-20 FPS

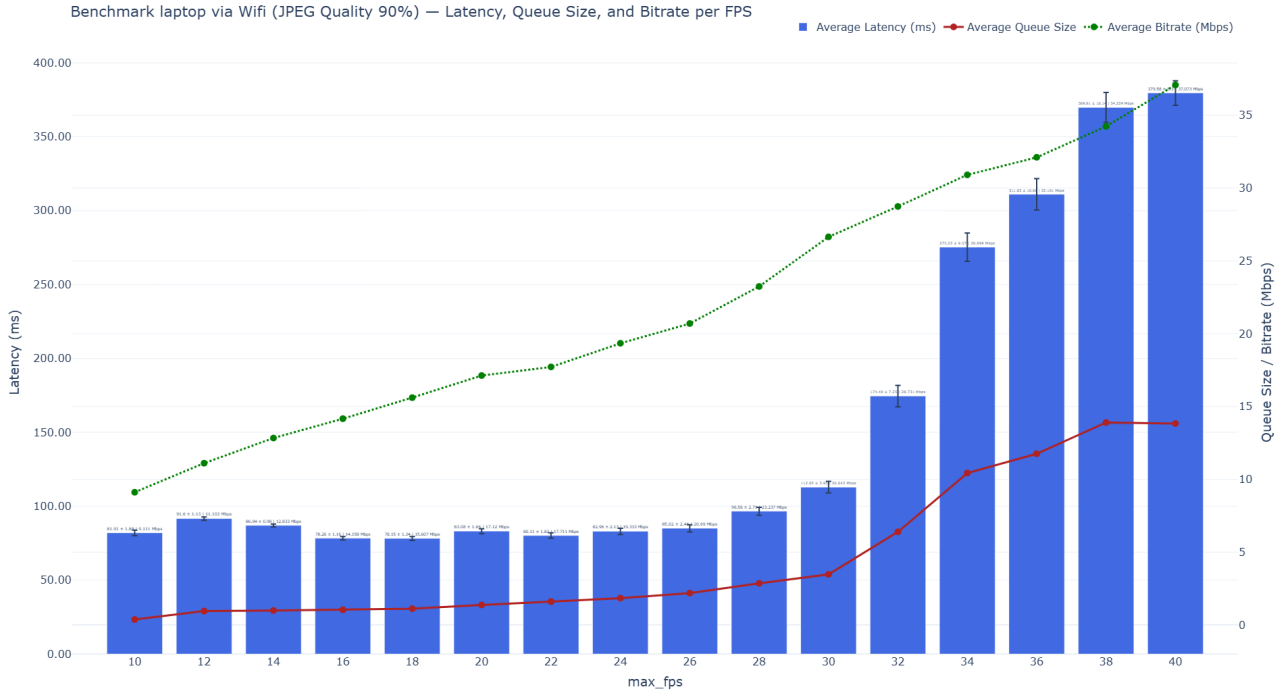


Figure 15: Benchmark Laptop Wi-Fi (JPEG quality 90%) real-time external video Inferencing (mean 95% CI)

but exceeded 200 ms from 24 FPS onward and surpassed 500 ms above 30 FPS, reaching approximately 700 ms at peak load.

Queue growth followed a similar pattern, becoming significant beyond 22 FPS and indicating that the inference server could not process incoming frames quickly enough at higher transmission rates. Despite throughput increasing almost linearly with FPS (up to 35 Mbps at 40 FPS), queuing delays dominated the end-to-end latency.

Overall, these results highlight that while domestic 5G provided sufficient capacity at low to moderate frame rates, stability deteriorated quickly once the system was stressed beyond 24-26 FPS. For safe assistive navigation at walking speed, the operational envelope should therefore be conservatively limited to 20-22 FPS on the laptop when using public 5G, ensuring latency remains below 200 ms and queue sizes do not compromise responsiveness.

6.1.2 Conclusions

As you can see on Figure 17, across all setups, the results confirm that real-time external inference is feasible on both Wi-Fi and domestic 5G, provided that the requested frame rate remains within a safe operational range. Local Wi-Fi offers the most stable behaviour, with low latency and negligible queue growth up to roughly 30 FPS. Domestic 5G introduces higher variability and an additional 30-70 ms delay, but still maintains end-to-end latency below the 200 ms walking threshold at moderate frame rates.

A consistent finding is that queue size is the earliest indicator of overload: once frames accumulate, latency rises sharply regardless of throughput. The RPI5 benefits from its lower compression throughput, which naturally limits ingress rate and delays queue formation compared to the laptop.

Overall, stable real-time performance is achieved between 20-28 FPS depending on network conditions. These results validate that cloud-assisted segmentation over public 5G can meet the responsiveness requirements for safe assistive navigation at walking speed.

6.1.3 Additional metrics

Processing overhead analysis

Processing overhead is shown in Figure 18 for four setups: a laptop on local Wi-Fi, a laptop on domestic 5G, a RPI5 on local Wi-Fi, and a RPI5 on domestic 5G. The graph displays compression time, encryption time, and the total processing overhead per frame. In all cases, the overhead share (percentage of total latency) remains below 1.9%, indicating that it has only a minor influence on the overall process.



Figure 16: Benchmark Laptop on 5G (JPEG quality 90%) real-time external video Inferencing (mean 95% CI)

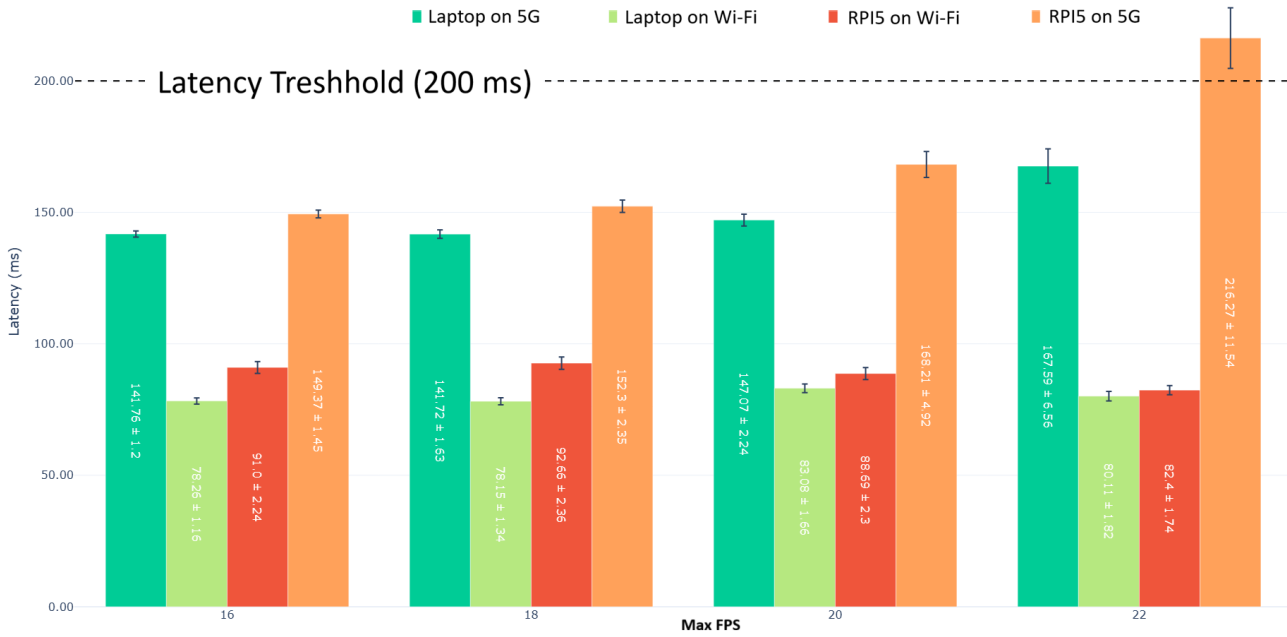


Figure 17: Benchmark compare Wi-Fi with 5G for real-time external video Inferencing (mean 95% CI)

The results demonstrate that compression dominates the total overhead, whereas encryption contributes only a negligible delay (<0.5 ms). On both laptop and RPI5, the total overhead remains between 2-4 ms. Laptops consistently require around 3.5-3.8 ms, while the RPI5 achieves slightly lower values of 2.3-3.5 ms. In relative terms, the overhead represents about 4 % of the total latency on laptops, whereas on the RPI5 it is closer to 1 %, since transmission and network delays dominate in that case.

Inference time and Frame level reliability

Figure 19 shows the relation between JPEG compression quality (10-100%) and both the average frame size (kB) and the frame-level success rate (%). Two clear trends are visible.

First, the average frame size increases almost linearly from approximately 15 kB at 10% JPEG quality to

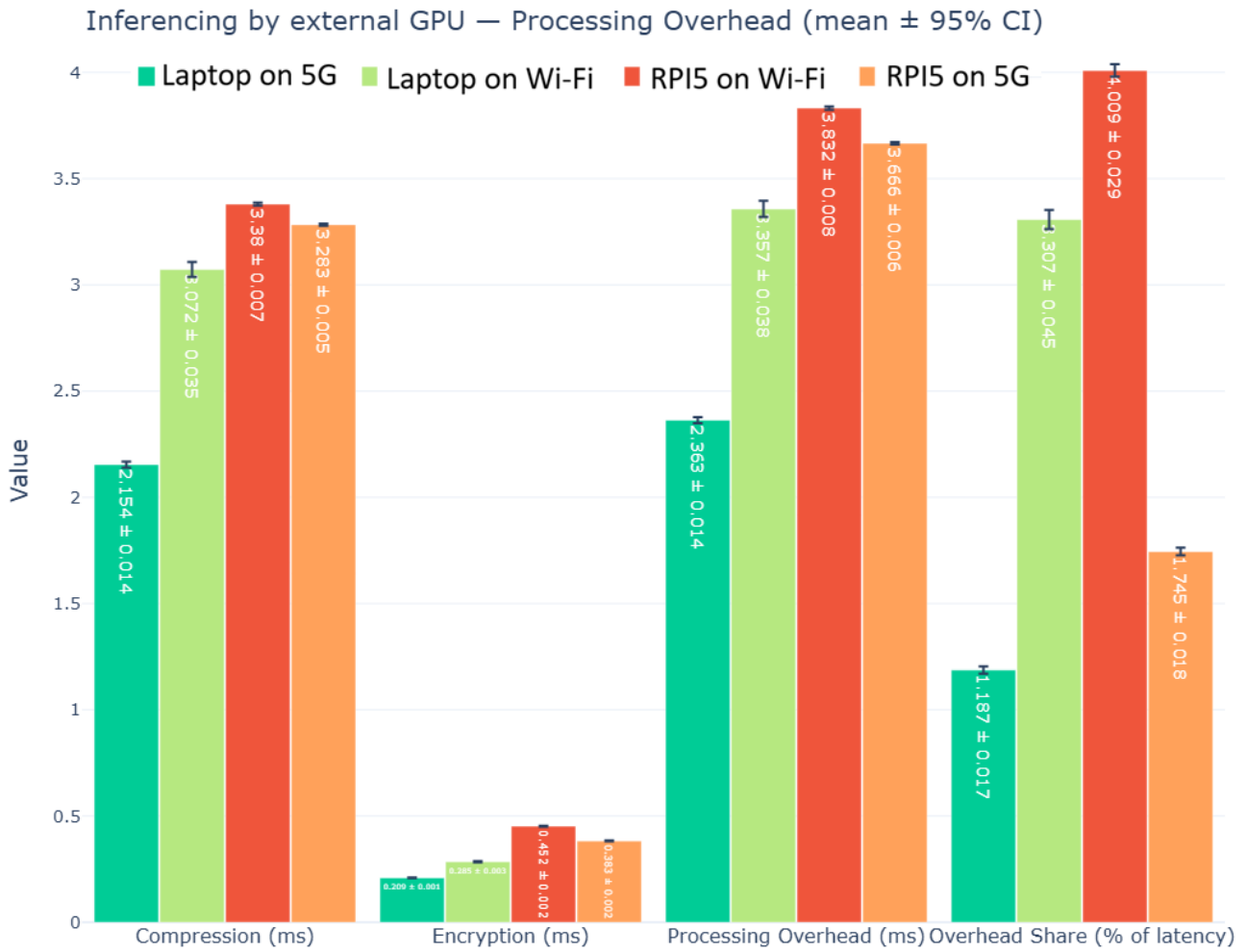


Figure 18: Processing overhead for real-time external video Inferencing Wi-Fi(DINNO) vs. 5G

110 kB at 90%. At 100% JPEG quality, a strong non-linear increase is observed, with an average size above 270 kB. This directly impacts the required transmission bandwidth and consequently the end-to-end latency.

Second, the frame-level reliability rises sharply at low qualities: from a reduced success rate at 10% to nearly 100% at 20%. Beyond this threshold, the success ratio remains stable and close to 100% for all higher qualities. This indicates that once a minimal visual quality is reached, further increasing JPEG quality does not significantly improve robustness.

From a system design perspective, this trade-off highlights that using medium JPEG quality settings (50–80%) provides the optimal balance. In this range, the success rate is saturated at nearly 100%, while the average frame size and bandwidth demand remain within acceptable limits. By contrast, very high quality (100%) imposes a large overhead without measurable gains in reliability. Therefore it is recommend to use 50–80% JPEG as the operating range.

Queue size

As discussed in Section 6.1.1, the queue size represents the number of frames waiting to be processed on the GPU inference server. It is a critical indicator of whether the requested frame rate exceeds the system’s real-time processing capacity. When the queue grows, latency increases proportionally, since each newly captured frame must wait for earlier ones to complete inference.

The results in the figures confirm this relationship. For the **RPI5 over 5G** (Figure 14), the queue size begins to rise from 24 FPS onwards, reaching values around 10 frames at 36–38 FPS. This correlates directly with the steep latency increase above 300 ms. In contrast, the **RPI5 over Wi-Fi** (Figure 13) shows much lower queue growth up to 30 FPS, but then exhibits a sharp increase beyond 34 FPS, where latency also spikes above 300 ms.

For the **laptop benchmarks**, a different pattern is observed. Over 5G (Figure 16), queue growth already starts at 20–22 FPS, and the queue stabilizes at high values (up to 20 frames) when the maximum FPS is pushed beyond 30. This leads to extreme latency values above 700 ms, showing that network-induced delays amplify

Inference time and Frame level reliability. JPEG Quality (10–100%)

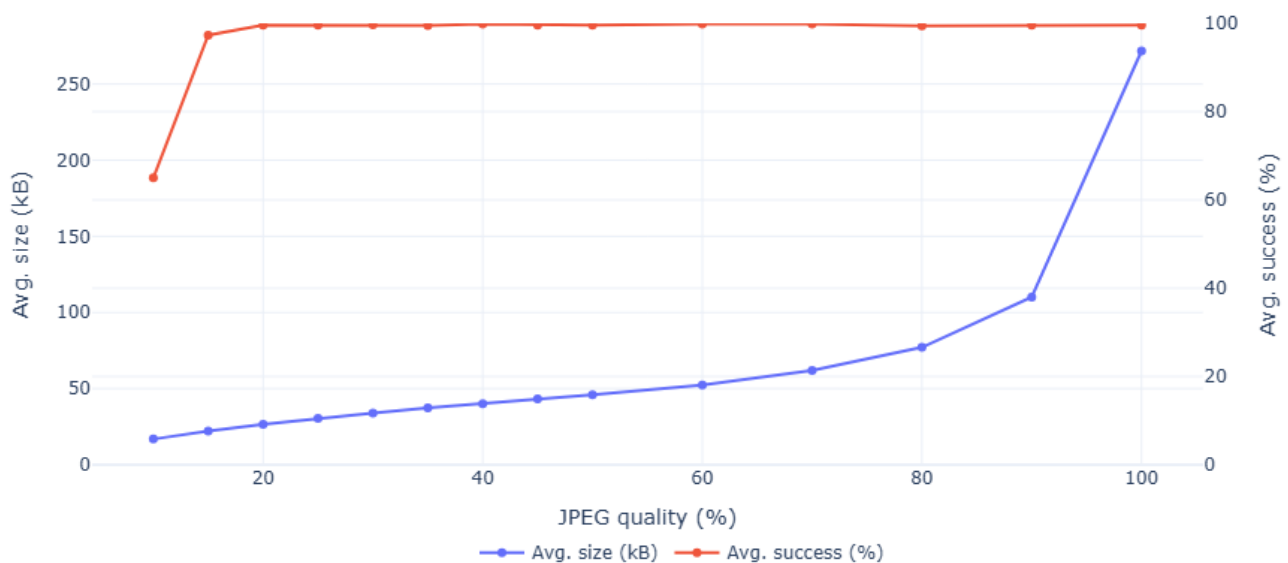


Figure 19: Inference time and Frame level reliability for real-time external video inferencing

the queuing effect. Over Wi-Fi (Figure 15), queue growth is delayed until 30 FPS, but when it occurs it rises steeply, again leading to significant latency increases.

Overall, these results highlight that the queue size is a strong early-warning indicator of system overload. Once frames accumulate in the queue, true real-time responsiveness is lost, even if average throughput (bitrate) appears sufficient. For safe assistive navigation, the operational envelope must therefore be chosen conservatively below the onset of queue growth.

Power consumption measurement

Power consumption on the laptop was measured using a Tuya 20A Zigbee smart power meter, integrated into the local Zigbee2MQTT network. The device provided updates at a frequency of approximately once per second, which was sufficient to obtain a reliable overview of average consumption during the benchmark runs. Although this temporal resolution does not capture very short-term spikes, it allows meaningful comparison of typical power usage across scenarios and provides insight into the overall energy demand of the system.

RPI5 power is currently an estimate based on CPU utilization scaling and should be interpreted qualitatively: Future work will include inline Direct Current(DC) measurements (INA219) to provide calibrated values and CIs.

This approach provides an approximate but practical measure of energy consumption without requiring external instrumentation, enabling lightweight evaluation of power trends during inference and streaming tasks.

Figure 20 compares the mean power consumption (poweruse_W) across different benchmark setups, expressed as mean values with 95% CIs. The results show a clear difference between laptops and RPI5 devices.

The **laptop benchmarks** demonstrate significantly higher consumption, with an average of 15.6 W on the domestic Wi-Fi connection and 10.7 W when connected via 5G. The **RPI5**, in contrast, consumes only 3.5 W on the domestic network and 2.5 W over 5G. This makes the RPI4/5 far more suitable for battery-powered, mobile use cases, where long runtime is essential.

Interestingly, the laptop consumes less energy on 5G than on the domestic Wi-Fi connection. This can be explained by the fact that in both cases the laptop connects through its Wi-Fi interface, while the actual 5G modem resides in the router. On the domestic Wi-Fi, higher throughput and lower jitter allow more frames to be transmitted and processed, keeping the CPU and GPU under a higher sustained load. In contrast, the 5G connection introduces lower throughput and higher variability, effectively throttling the data pipeline. This reduces the number of frames processed per unit time, and thereby lowers the average system power consumption of the laptop.

The **key insight** is that network conditions not only affect latency and frame reliability, but also indirectly influence power usage on the end-device. In practice, this means that lower link quality may appear to save energy, but at the cost of reduced effective frame rate and degraded real-time responsiveness.

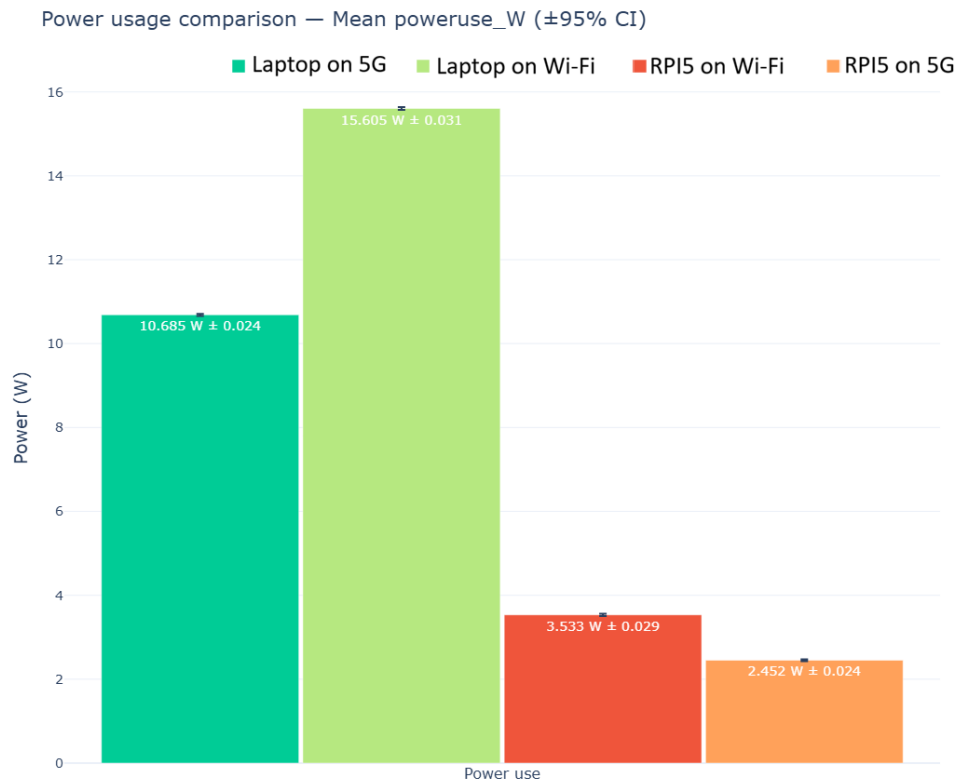


Figure 20: Estimated power draw (W) by device/network, mean $\pm$ 95% t-CI (proxy-based for RPI)

6.1.4 Conclusions

The additional metrics confirm that processing overhead, inference time, queue growth, and power usage each play a distinct role in overall system behaviour. Processing overhead remains negligible across all setups, with compression dominating and encryption contributing only sub-millisecond delay. Frame-level reliability stabilises once a minimal JPEG quality is reached, indicating that medium compression levels (50-80%) provide the best trade-off between bandwidth and robustness. Queue size again emerges as a key indicator of overload, once it begins to rise, latency increases sharply and real-time responsiveness degrades. Power measurements highlight the suitability of the RPI5 for mobile use, while also showing that network conditions indirectly affect energy consumption. Overall, these complementary metrics reinforce the importance of operating within conservative FPS and bitrate limits to maintain stable real-time performance.

6.2 Real-time local inference on edge-devices

Measuring end-to-End latency when running the inference task on the device itself is different, as stated in Section 3.2, is now calculated based on the inference time and the processing time.

We should also make sure that the general load of other tasks running on the end-device is as low as possible by closing all unnecessary tasks and processes running on that device.

The results of this experiment on different end devices are shown in Figure 21. To improve readability, a logarithmic (log) scale was used in the graph.

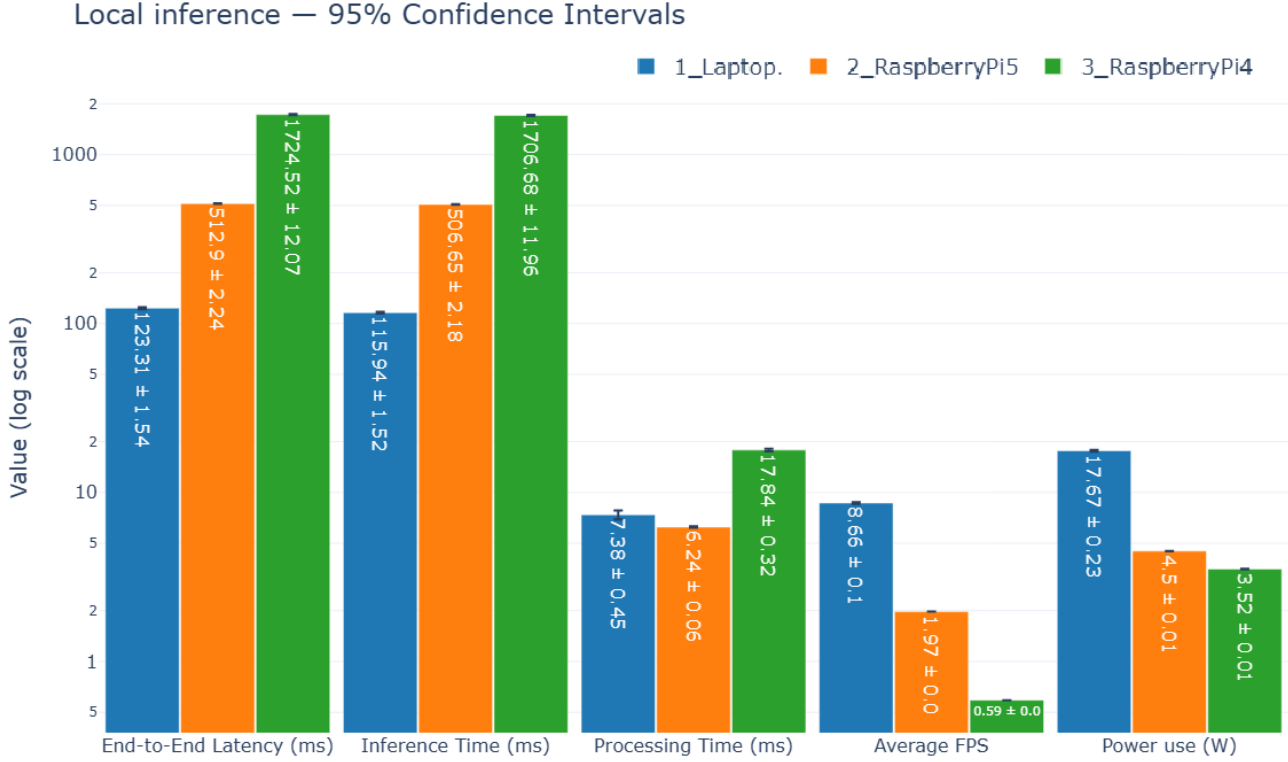


Figure 21: Local inference on different end-devices (log scale)

6.2.1 Standard laptop

As shown in Figure 21, local inference was first benchmarked on a standard laptop equipped with a CPU but without a dedicated GPU. The average end-to-end latency was 123.31 ms, with an inference time of approximately 115.9 ms per frame. This corresponds to an average frame rate of 8.66 FPS, meaning that the laptop can process roughly nine images per second without relying on any external network. These values are sufficient for near real-time performance in some applications, since the total latency remains below 200 ms, which is often considered the upper limit for human-perceived real-time feedback.

In terms of power consumption, the laptop required about 11.0 W when idle and 17.67 W during inference, representing an increase of 6.67 W due to CPU utilization. While these results confirm that local inference on a CPU-based laptop is feasible, the setup is less energy-efficient and less portable compared to lightweight Edge devices such as the RPI series.

6.2.2 RPI4 and RPI5

The same local inference benchmark was performed on both a RPI4 and a RPI5 to assess the balance between performance and energy efficiency on embedded systems. On the RPI4, the measured end-to-end latency was approximately 1.7 s per frame, resulting in an average frame rate of only 0.59 FPS. This clearly shows that the RPi 4 is unsuitable for real-time guidance tasks, as visual feedback would lag several seconds behind user movement.

The RPI5, featuring a faster processor and upgraded architecture, performed significantly better with an end-to-end latency of 512.9 ms and a throughput of 1.97 FPS. Although this represents a major improvement over the RPI4, it still falls short of the 10–15 FPS typically required for smooth and continuous visual guidance. The main advantage of the RPI devices lies in their low power consumption: the RPI5 consumed only 4.5 W during inference and less than 5 W in total, compared to 17.67 W for the laptop.

These results highlight the trade-off between processing performance and energy efficiency in local inference. While the laptop achieves lower latency, its higher power usage and limited portability make it less suitable for mobile assistive navigation. Conversely, the RPI devices are compact and energy-efficient but currently lack the computational capacity for real-time operation, motivating the exploration of edge and cloud-assisted inference in later sections.

6.3 Stability of latency and Link Quality over 4/5G Network Setup

6.3.1 Objectives

As discussed in Section 1.2.1, it is essential to analyze the quality of the 4/5G connection to ensure that the system maintains acceptable latency, frame rate, and link stability to support real-time path planning during outdoor activity, where radio conditions are time-varying.

This experiment aims to characterize the variability of the public 4/5G connection over a representative park environment (Laerbeekbos, Brussels) and to determine whether these conditions permit reliable cloud-assisted segmentation inference at walking speeds. The test was conducted on September 23, 2025, at 13:00 under partly cloudy conditions, using the Orange 5G NR-NSA network via a Netgear Nighthawk M6 router. In total 25 measurement points were recorded at different points of the walking trail.

This park was selected because of its good 5G coverage, mixed open and wooded areas, and its realistic representation of semi-urban park environments where assistive navigation systems for VIPs could be deployed.

6.3.2 Edge-device hardware setup (RPI5)

The edge-device is a RPI5 with 8 GB RAM. For field monitoring, the Pi drives Epson BT-40 AR glasses via its High-Definition Multimedia Interface (HDMI) video output, enabling live inspection of telemetry (e.g., logs, FPS, and latency) while walking test routes without a laptop. Position data are captured by an Adafruit Ultimate GPS via USB: The 66-channel receiver provides fixes up to 10Hz that are time-aligned with the video stream and inference results for later analysis and mapping. The system is powered by a Bluebuilt 27000 mAh / 100W power bank (model BBLP2710S), supporting several hours of fully mobile operation.

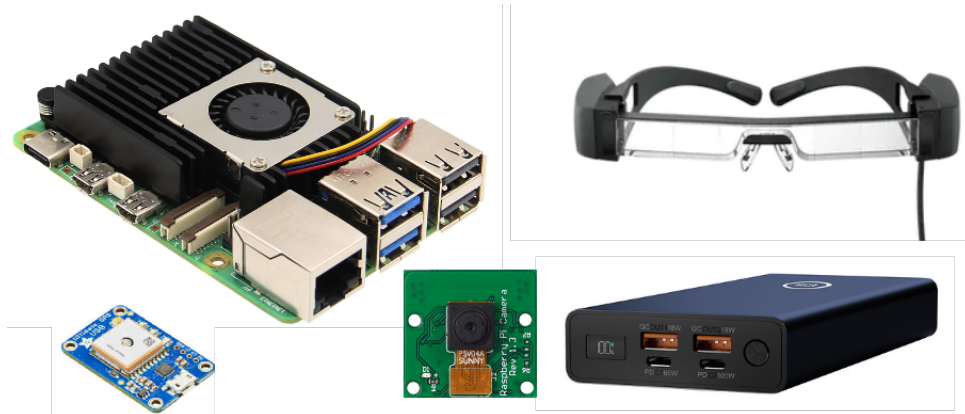


Figure 22: raspberrypi 5 with Epson BT-40, RPICAM, GPS and batterypack

Please note that for the benchmarks in this thesis, a camera is not used, as we use a prerecorded video. The components can be seen in Figure 22.

6.3.3 Measurements

At each GPS coordinate, an Ookla Speedtest measurement was performed to obtain the download and upload throughput, latency and jitter. 25 Samples were taken at approximately 1-minute intervals while walking along the main path, covering both open and shadowed areas.

Figure 23 visualizes the measurement results on an OpenStreetMap background. Marker size corresponds to the measured download speed (in Mbps), while color represents latency (in ms). Dark purple indicates low latency (below 40 ms), whereas yellow to orange shades indicate increased delay. The figure demonstrates that latency remained consistently low across most of the park.

The map was generated using Python with the Folium and Pandas libraries, which read GPS-tagged Ookla measurement data and dynamically adjusted marker size and color scales. This visualization method allows an intuitive spatial understanding of network behavior along the test route.

A summary of the statistical results is presented in Table 6. Values are shown as minimum, maximum, and mean across all samples.

Across all samples, the mean latency was approximately 24.4 ms, with 95% of the values below 38 ms. Average download and upload throughput were 19.6 Mbps and 5.1 Mbps, respectively, and jitter remained below 5 ms for most locations. These results demonstrate that public 5G connections can sustain real-time streaming of compressed video for AI segmentation tasks.

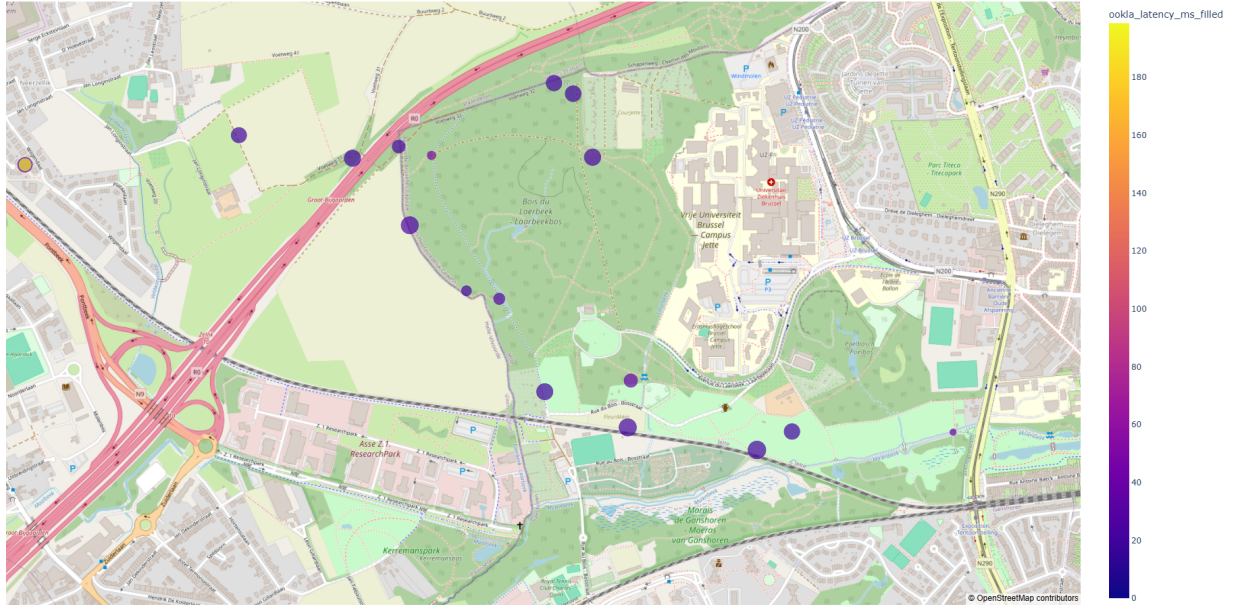


Figure 23: Ookla field test results in Laerbeekbos: marker size = download rate, color = latency

Table 6: Summary of Ookla results (min, max, mean)

	Download (Mbps)	Upload (Mbps)	Latency (ms)	Jitter (ms)
min	4.49	1.35	19.01	0.08
max	28.03	13.35	41.45	15.48
mean	19.63	5.08	24.41	4.02

6.3.4 Conclusions

The field measurements confirm that public 5G NR-NSA networks provide sufficient stability for real-time assistive navigation in outdoor park environments. Median latency of around 21 ms and low jitter (2–4 ms) indicate that end-to-end latency will remain well below the 200 ms threshold considered safe for haptic feedback and walking guidance.

Only one measurement instance experienced a fallback to 4G LTE, characterized by a download speed of 4.5 Mbps, an upload of 1.4 Mbps, and a recorded latency of 41 ms for this measurement. Although still usable, this illustrates the importance of continuously monitoring link state to trigger adaptive bitrate control and local failsafe mechanisms when needed.

Overall, the observed throughput and latency values are sufficient for real-time cloud-assisted segmentation, provided that the uplink bitrate remains below approximately 3–4 Mbps and local safety mechanisms (e.g., smart cane or onboard depth sensors) remain active. These findings confirm that the proposed system architecture can operate reliably over existing public 5G infrastructure, making large-scale deployment feasible without requiring private network slicing.

7 Conclusions

This thesis evaluated the feasibility of a real-time, vision-based assistive navigation system for VIPS using public 4G/5G networks. Extensive benchmarks compared edge-based inference on devices such as the RPI4/5 and laptop with cloud-based GPU inference, analyzing key performance indicators such as true end-to-end latency, frame rates, power consumption, inference time, frame-level reliability, and queue dynamics.

The results demonstrate that external GPU inference over domestic 5G networks can maintain end-to-end latencies easily below 200 ms, which is acceptable for safe navigation at walking speed. While edge devices such as the RPI5 are energy-efficient and portable, they lack sufficient computational capacity for real-time segmentation. Cloud-based inference therefore remains essential, provided that robust fallback mechanisms (e.g., smart cane, local obstacle detection) are in place. The tests further confirm that Wi-Fi connections yield more stable performance, yet public 5G links still deliver adequate responsiveness for real-world deployment, validating the viability of this approach.

Power consumption benchmarks show that low-power devices such as the RPI5 (2.5–3.5 W) are far better suited for wearable use compared to laptops (10–17 W). Queue analysis highlighted the importance of selecting conservative FPS settings to avoid instability, while inference times around 12 ms on the GPU server confirm real-time capability even on lower-end hardware.

An additional consideration is the economic feasibility of deployment. Assuming an average bitrate of 7 Mbit/s, while continuous streaming for one hour takes approximately 3.15 GB of mobile data use. With a domestic subscription of 15 GB/month at EUR 5, this corresponds to nearly five hours of guided walking per month. This cost structure illustrates that the solution is not only technically viable but also affordable for end users.

This thesis establishes the scientific and technical foundation for the *BlindNavParks* project, demonstrating that real-time, cloud-assisted navigation for visually impaired persons over public 5G is both feasible and cost-effective. The results directly support the implementation of pilot trials funded by the Koning Boudewijnstichting, paving the way for inclusive, scalable navigation solutions in public parks.

8 Ongoing developments and Future work

As a researcher affiliated with the EHB Research Centre for Preventive Health Innovation, my work aligns with the research line “Human-centered AI and robotics technologies for decision-making and training.” Within this context, the following initiatives represent ongoing developments that build upon the results of this thesis, as well as directions for future research and application.

8.1 Proof-of-Concept: Real-Time Interactive Simulation of a VIP in Unreal Engine

Objectives

The objective of this proof-of-concept (POC) is to demonstrate a safe, interactive setup in which a robotic platform simulates the actions of a visually impaired person (VIP) navigating through a park. Unlike the prerecorded video sequences used in earlier benchmarks (Section 3.2), this setup integrates a live Unreal Engine environment whose rendered output serves as the real-time video source for the segmentation and guidance pipeline.

The goal is twofold: (1) to validate the end-to-end streaming and feedback loop in an interactive scenario, and (2) to offer a controllable, laboratory-safe environment in which VIP users can later train or experience the haptic feedback system without the risks of outdoor navigation.

Setup and Methodology

The experimental setup consists of a PC running Unreal Engine 5.6. The screen output is used as input stream for the inference pipeline.

A compact custom designed robot platform (Based on Turtlebot3 components) equipped with a gripper that physically holds a standard computer mouse (Figure 24). This robot functions as a VIP simulator.

The Unreal Engine game simulates a first-person view (FPV) of a person walking along a virtual park trail. The rendered game screen is captured in real time and streamed through the same video segmentation and feedback pipeline as used in the benchmarks (Section 6.1). However, in this case, no prerecorded video is used: Each frame is generated on the fly by the Unreal Engine scene.

The robot effectively controls the camera direction in the Unreal Engine simulation by moving the mouse left or right, mimicking the head or walking direction choices a VIP would make in a real environment. Forward motion in the virtual park is controlled manually by the user through keyboard input, which represents the walking pace of a VIP.

The feedback loop is identical to the real-world configuration: Segmented path information is processed by the GPU inference server and converted into directional feedback signals. These signals could later be mapped to haptic cues, allowing the user to experience steering corrections or obstacle warnings through tactile output.

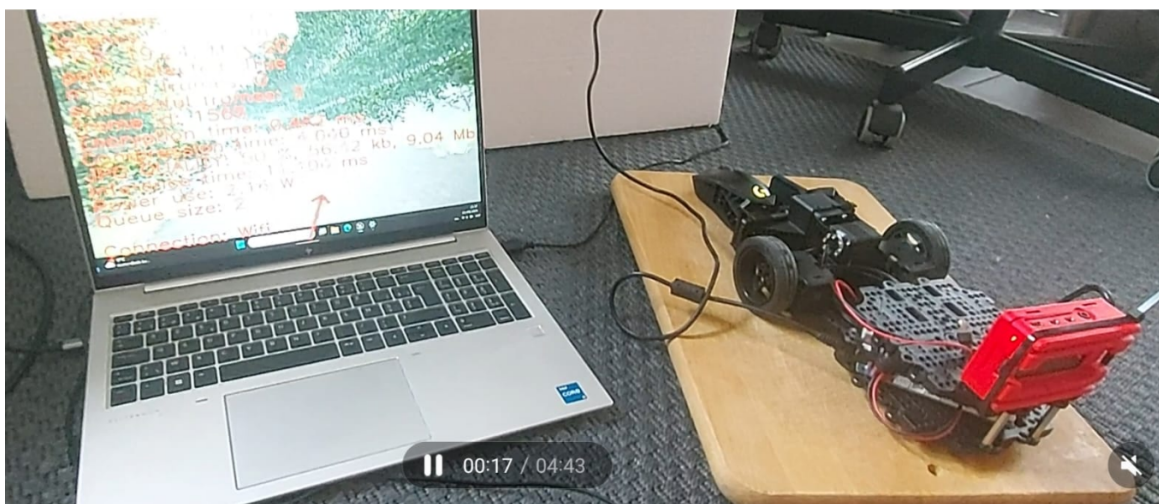


Figure 24: Interactive proof-of-concept: robot controlling Unreal Engine simulation by moving a mouse

Results and Applications

The setup successfully demonstrated stable end-to-end streaming and control at interactive frame rates. Latency remained within the same operational range as during the prerecorded tests, confirming that the segmentation

and feedback loop can operate in real time even with dynamically generated frames.

This simulation method provides a valuable intermediate stage between laboratory development and field testing. It allows safe evaluation of different feedback modalities and control parameters while maintaining a realistic navigation experience. In future user studies, the same setup can serve as a training environment for VIP participants to familiarize themselves with haptic cues and steering feedback before engaging in outdoor trials.

8.2 Smartphone Prototype: *Stay on Trails*

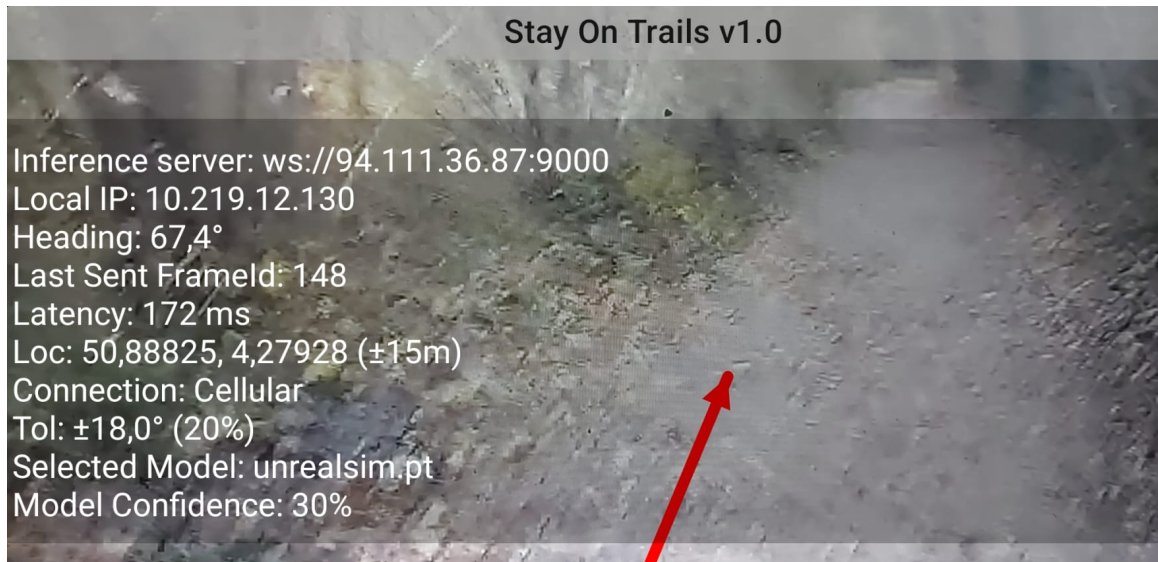


Figure 25: Visualisation Android app : Stay on trails

As part of the ongoing developments toward practical and user-friendly assistive navigation tools, a dedicated smartphone prototype *Stay on Trails* [10] was developed to evaluate whether the core concepts of real-time path segmentation and heading computation can operate effectively on widely available mobile devices. The application runs on an Android smartphone and implements a lightweight, real-time client that streams camera frames to a remote GPU server for segmentation. See Figure 25.

System architecture and network pipeline The app streams live camera frames from the smartphone to a remote segmentation server using either Wi-Fi or a mobile 5G connection. Each frame is compressed (JPEG, approximately 50 % quality), timestamped, and transmitted via a persistent WebSocket connection to a publicly accessible signaling server (port 9000).

Camera acquisition and frame preparation The application uses the Android CameraX library to capture 640×480 YUV420 frames. Each frame is processed through a consistent preparation pipeline:

1. conversion from YUV420_888 to NV21;
2. JPEG compression using `YuvImage` at approximately 50 % quality;
3. enforcement of a minimum inter-frame interval (default 200 ms) to control bandwidth usage and server load.

WebSocket communication and telemetry The `WsJpegUploader` component manages the WebSocket session. For every frame, a compact JSON metadata object is transmitted first, containing:

- frame ID and timestamp,
- network connection type (WiFi / Cellular),
- last measured latency,
- GPS coordinates,
- selected AI model or recording mode.

The binary JPEG data is transmitted next. When the GPU server responds with the computed heading, the smartphone calculates the end-to-end latency by comparing the current time to the locally recorded timestamp associated with the frame ID. This mechanism aligns with the frame-ID-based latency methodology used throughout the main thesis.

Visual and auditory feedback The computed heading is communicated to the user through two complementary channels:

1. **Visual:** the custom `ArrowOverlay` view draws a red arrow extending from the bottom center of the screen toward the recommended walking direction.
2. **Auditory:** the continuous heading angle is discretized into three states (**LEFT**, **FORWARD**, **RIGHT**) based on a configurable tolerance. When a direction change is detected, the corresponding spoken cue (**left**, **right**, **forward**) is played, preventing unnecessary auditory overload.

A real-time telemetry overlay displays additional information such as server address, latency, GPS position, selected model, and confidence score, supporting both debugging and field evaluation.

Configurable parameters and recording mode A dedicated settings interface (`SettingsActivity`) allows the user to adjust several parameters: minimum interval between transmitted frames, heading tolerance for direction classification, minimum model confidence, chosen segmentation model (e.g., `unrealsim.pt`, `canicross.pt`) or *recording mode*.

In recording mode, transmitted frames are also saved server-side, enabling the construction of new training datasets for park-specific segmentation models. This capability supports the long-term goal of developing models tailored to diverse outdoor environments.

GPS telemetry and real-world evaluation The app periodically sends GPS updates to the server. Combined with latency and connection-type data, these measurements allow the full system performance to be analyzed both in controlled lab settings and in real outdoor environments. For example, the latency along a walk in Laerbeekbos was clustered into color-coded intervals (green to red), revealing spatial variations in responsiveness and helping to identify areas where user feedback may degrade. The results of a test seen on Figure 26 with color indications, green for max 140 ms, lightgreen of 140 to 170 ms, oranje for 170 to 210 ms and red for 210 to 260 ms.

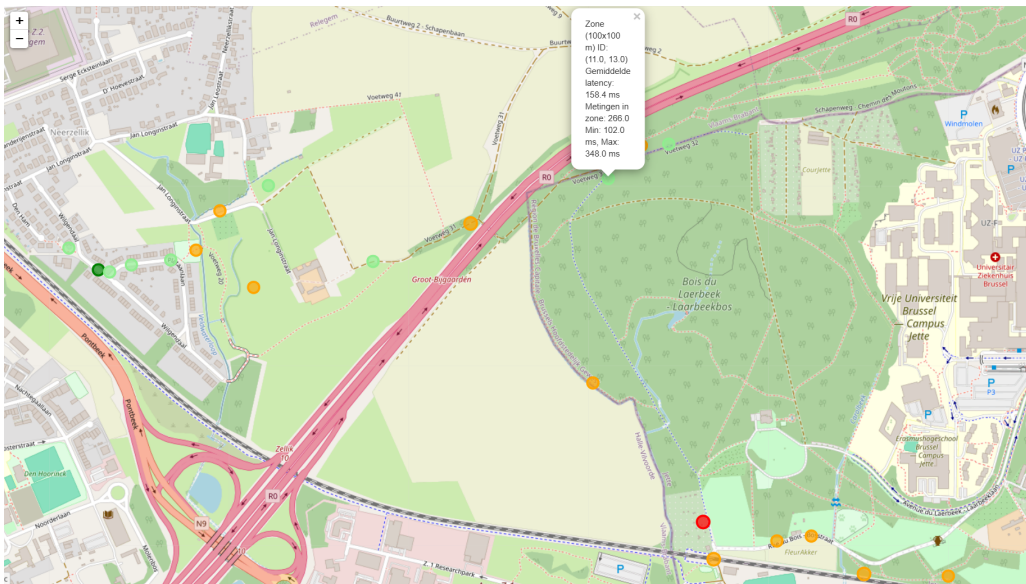


Figure 26: Visualisation Stay on trails while walking in Laerbeekbos on 07/12/2025

Relevance to this thesis The *Stay on Trails* prototype demonstrates that the architecture validated in this thesis, remote segmentation, row-wise midpoint heading computation, and frame-ID-based latency measurement can operate on a consumer smartphone without additional hardware. This confirms the feasibility of transitioning from RPI-based experimental platforms to a smartphone-centered, accessible solution for visually impaired users.

8.3 BlindNavpark(Koning Boudewijnstichting funding request)

The *BlindNavParks* initiative, submitted on 27/06/2025 by Erasmushogeschool Brussel, has been approved on 19/09/2025 by the Koning boudewijnstichting under the call *Visual Impairment & Inclusion* [20]. The project received a subsidy of EUR 20 000 to support a two-year pilot phase (2025-2027) in two Brussels parks.

The funding will primarily be used to develop a user-friendly interface and mobile optimizations on top of the existing segmentation pipeline. Additionally, it covers server costs for hosting, storage, and model training, as well as test hardware such as RPI devices, cameras, battery packs, and haptic feedback interfaces.

BlindNavParks aims to enable blind and VIPS to walk safely and autonomously in parks. Camera images are processed through AI-based segmentation to recognize the path, after which the safe walking direction is provided via haptic feedback. Volunteers, accessibility organizations, and local stakeholders are involved in supporting the annotation of walking routes and user testing. The project thus translates advanced AI research into a practical, inclusive solution for real-world navigation support.

This master thesis provides the scientific background and benchmarks necessary to demonstrate the technical feasibility of such an approach, thereby forming the basis on which the BlindNavParks project can be realized successfully.

8.4 Future Work

Key exchange mechanism: While the prototype implementation secures video frames using AES-256 in CBC mode, future versions of the system should migrate to a fully standardized transport-layer security (TLS) layer. A natural evolution is to encapsulate all communication, including video frames, metadata, and control messages, within a TLS-encrypted WebSocket channel. TLS 1.2/1.3 provides authenticated encryption by default (typically using Authenticated Encryption with Associated Data(AEAD) ciphers such as Advanced Encryption Standard, Galois/Counter Mode(AES-GCM) or ChaCha20-Poly1305), thereby ensuring both confidentiality and integrity of the transmitted data. Moreover, TLS enables well-established mechanisms for certificate-based authentication and forward secrecy, making it more suitable for deployment in real-world, security- and privacy-sensitive assistive navigation systems.

VIP participants: Although this study relies on robotic simulators and prerecorded FPV sequences to ensure safety and reproducibility, the real effectiveness of the system must ultimately be validated with VIPS. Future work should therefore include structured trials with blind users to evaluate the perceived real-time responsiveness of the navigation aid. Key aspects include measuring the subjective experience of latency, the clarity of haptic feedback, and the overall trust in the system under real walking conditions. Such user-centered evaluation will provide essential insights into usability, acceptance, and potential refinements that cannot be captured in simulation alone. Ethical approval, safety protocols, and close involvement of accessibility organizations will be required to conduct these tests responsibly.

Implementation on smartphone: An important next step is the implementation of the navigation pipeline directly on smartphones. Unlike dedicated edge devices such as RPI, smartphones already integrate high-quality cameras, built-in 4G/5G connectivity, and efficient AI accelerators. This makes them an ideal platform for large-scale adoption, as no additional hardware is required beyond a mobile application. A smartphone-based implementation would lower the entry barrier for VIPS, enabling deployment through existing devices they already carry.

Adaptive Bitrate: While the present thesis fixes video compression quality during benchmarks to ensure comparability, a practical deployment would benefit from an adaptive bitrate (ABR) strategy. ABR dynamically adjusts the encoding quality of video frames based on current network throughput and latency, thereby avoiding buffer growth and excessive queuing during temporary link degradation. In the context of assistive navigation, ABR could ensure that critical path information is still transmitted at lower quality when bandwidth is constrained, rather than risking frame drops or unsafe latency. Future work should therefore investigate ABR mechanisms tailored to mobile 4G/5G conditions, with a focus on maintaining frame-level reliability and safe end-to-end responsiveness, while balancing energy consumption on the edge device.

GDPR Human trials with VIPs will require ethics approval, an escort protocol, DPIA/GDPR compliance, and explicit consent. This includes that no raw video will be stored: Only frame-level telemetry (timestamps, latency, success) will be retained for 30 days for debugging.

9 Word list

- Adaptive Compression = Dynamic adjustment of video compression level to balance visual quality and transmission bandwidth.
- CBC with PKCS#7 = Cipher Block Chaining; a mode of operation for block ciphers such as AES, where each plaintext block is XORed with the previous ciphertext block before encryption, providing confidentiality. PKCS#7 is a padding scheme ensuring that the final plaintext block matches the cipher's block size.
- Cloud Inference = AI processing performed on a remote GPU server rather than on the local device, enabling higher performance but requiring stable network connectivity.
- Edge Device = A lightweight computing platform (e.g., Raspberry Pi, smartphone) used near the data source for capturing and transmitting frames.
- Encryption (AES-256 CBC) = Advanced Encryption Standard with 256-bit key in Cipher Block Chaining mode; used for securing video data.
- ESPNet = Efficient Spatial Pyramid Network; a lightweight convolutional neural network for real-time semantic segmentation, using a spatial pyramid of dilated convolutions to efficiently capture both local and global context.
- Failsafe Mechanism = A backup system (e.g., bumper sensors, smart cane) ensuring safety when the AI system or network connection fails.
- Haptic Feedback = Signals perceived through touch or vibration rather than vision or sound.
- Inference Time = The time taken by an AI model to process an input frame and produce an output result.
- Jitter = The variation in latency over time; high jitter negatively affects real-time responsiveness.
- Packet Loss = The percentage of data packets that fail to arrive at the destination, reducing reliability.
- Queue Size = The number of frames waiting to be processed by the inference server; a key indicator of system overload.
- Segmentation = A computer vision task of dividing an image into regions (e.g., path vs. non-path) to guide navigation.
- Smart Cane = An assistive mobility device with sensors that detect obstacles at knee level or higher, providing a local failsafe layer.
- Throughput = The actual data transfer rate over a network, measured in Mbit/s, determining how many frames per second can be transmitted.
- True end-to-end Latency = The total time between capturing a video frame on the edge device and receiving processed feedback from the inference server.

10 Appendix

10.1 TCP - Transport layer protocol

The Transmission Control Protocol (TCP) [31] is the primary transport layer protocol in the TCP/IP stack, enabling reliable, ordered, bidirectional byte-stream communication between computers. It incorporates flow control and congestion control mechanisms, with the latter being critical to maintaining the stability and functionality of the Internet.

10.2 Websocket application layer protocol

The WebSocket Protocol [11] is an application-layer protocol built over TCP that enables bidirectional communication between client and server. Unlike traditional HTTP, which relies on a pull-based model where only the client initiates interactions, WebSocket allows the server to send unsolicited messages to the client. This approach consolidates communication into a single persistent TCP connection, reducing the need for multiple simultaneous connections. Additionally, WebSocket minimizes HTTP header overhead, thereby improving bandwidth efficiency.

10.3 WebRTC

WebRTC [37] is an API standard that enables peer-to-peer connections between browsers over IPv4, IPv6, and dual-stack networks. It is designed to maintain performance under challenging conditions such as wireless or congested links. WebRTC allows direct, encrypted transmission of video, audio, and text between peers. NAT traversal is achieved through STUN, TURN, and ICE mechanisms, while a signaling server facilitates peer discovery and address exchange. Media streams are transported using Secure Real-Time Transport Protocol (SRTP) over DTLS, and text data is transmitted via Data Channels employing SCTP over DTLS. DTLS is mandatory, and a single secure connection handles both media and data traffic.

10.4 Network Address Translation (NAT)

Network Address Translation (NAT) [15] is employed in this project to enable communication between devices on private local networks and external systems over the public Internet. The 5G router and the domestic cable or fiber router both operate as NAT gateways. Outgoing connections from the sender (the end device) are translated from a private IP address to a public IP address assigned by the mobile carrier or Internet service provider. This allows the signaling server to receive and respond to requests transparently.

To permit inbound connections to the signaling server hosted on the local network, explicit port forwarding is configured (e.g., port 9000) so that external clients can initiate WebSocket sessions. This approach ensures that devices behind NAT remain reachable for signaling while maintaining a secure separation between internal and public networks. NAT thus plays a central role in enabling reliable, bi-directional data exchange in the experimental setup.

11 Bibliography

References

- [1] .lumen. *The .lumen Glasses: Artificial Intelligence that Empowers the Blind*. Accessed: 5 October 2025. 2025. URL: <https://dotlumen.com>.
- [2] Tommy Azzino et al. “5G Edge Vision: Wearable Assistive Technology for People with Blindness and Low Vision”. In: *arXiv preprint arXiv:2311.13939* (2023). URL: <https://arxiv.org/abs/2311.13939>.
- [3] Belgian Institute for Postal Services and Telecommunications (BIPT). *5G roll-out in Belgium: network mapping and quality*. Accessed: 16 September 2025. Mar. 2024. URL: <https://www.bipt.be/consumers/publication/5g-roll-out-in-belgium-network-mapping-and-quality>.
- [4] László Blázovics, Viktor Kovács, and Márton Gótzty. “Low Latency Video Streaming System for VR Teleoperation over 5G Networks”. In: *2024 IEEE International Conference on Metrology for eXtended Reality, Artificial Intelligence and Neural Engineering (MetroXRINE)*. 2024, pp. 982–987. DOI: [10.1109/MetroXRINE62247.2024.10796131](https://doi.org/10.1109/MetroXRINE62247.2024.10796131).
- [5] Y. Chen, L. Wang, and F. Zhao. “AI Guide Dog: Egocentric Path Prediction on Smartphone”. In: *arXiv preprint arXiv:2501.07957* (2025). arXiv: [2501.07957](https://arxiv.org/abs/2501.07957). URL: <https://arxiv.org/abs/2501.07957>.
- [6] Sinuk Choi, Dongyoon Kwon, and Ji-Woong Choi. “Latency Analysis for Real-Time Sensor Sharing Using 4G/5G C-V2X Uu Interfaces”. In: *IEEE Access* 11 (2023), pp. 35197–35206. DOI: [10.1109/ACCESS.2023.3265474](https://doi.org/10.1109/ACCESS.2023.3265474).
- [7] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer, 2002. ISBN: 978-3-540-42580-9.
- [8] Maarten Dequanter. *5GWebRtc: Evaluation of WebSocket and WebRTC Communication for Real-Time Segmentation*. <https://github.com/mdequanter/5GWebRtc>. Accessed: 2025-07-05. 2025.
- [9] Maarten Dequanter. *BlindNav: Haptic Braille Feedback Prototype on Tinkercad*. <https://www.tinkercad.com/things/10EKNeYcI3J-blindnav>. Online prototype demonstration. Accessed: 6 Oct. 2025. 2025.
- [10] Maarten Dequanter. *Stay on Trails: Android Application for Real-Time Trail Navigation*. <https://github.com/mdequanter/BlindNavApp>. Accessed: 2025-12-08. 2025.
- [11] I. Fette and A. Melnikov. *The WebSocket Protocol*. RFC 6455. <https://datatracker.ietf.org/doc/html/rfc6455>. Dec. 2011. DOI: [10.17487/RFC6455](https://doi.org/10.17487/RFC6455). URL: <https://datatracker.ietf.org/doc/html/rfc6455>.
- [12] Jingao Gao et al. “Edge Assisted Real-time Object Detection for Mobile Augmented Reality”. In: *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services (MobiCom '19)*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 420–433. ISBN: 978-1-4503-6661-4. DOI: [10.1145/3300061.3300116](https://doi.org/10.1145/3300061.3300116).
- [13] Xinyu Gao, Yuanwei Liu, and Xidong Mu. “SLARM: Simultaneous localization and radio mapping for communication-aware connected robot”. In: *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2021, pp. 1–6.
- [14] Tadeh Ghahremanians and Hossein Mahvash Mohammadi. “Efficient Real-Time Pathfinding for Visually Impaired Individuals”. In: *IEEE Access* 13 (2025), pp. 71323–71334. DOI: [10.1109/ACCESS.2025.3562247](https://doi.org/10.1109/ACCESS.2025.3562247).
- [15] M. Holdrege and P. Srisuresh. *IP in IP Tunneling with Network Address Translation (NAT)*. RFC 4008. Mar. 2005. DOI: [10.17487/RFC4008](https://doi.org/10.17487/RFC4008). URL: <https://datatracker.ietf.org/doc/rfc4008/>.
- [16] I-Cane. *I-Cane Mobilo: Smart Cane for the Visually Impaired*. Accessed: 5 October 2025. 2025. URL: <https://i-cane.nl/home>.
- [17] Admir Kaknjo et al. “Real-Time Video Latency Measurement between a Robot and Its Remote Control Station: Causes and Mitigation”. In: *Wireless Communications and Mobile Computing* 2018.1 (2018), p. 8638019.
- [18] Robert K Katzschnmann, Brandon Araki, and Daniela Rus. “Safe local navigation for visually impaired users with a time-of-flight and haptic feedback device”. In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 26.3 (2018), pp. 583–593.
- [19] Abdullah Khan et al. “Enhancing Smart Outdoor Object Navigation for the Visually Impaired via YOLOv10 with Neighbor Coordinates and C2FCIB Attention Mechanism”. In: *IEEE Access* (2025).

- [20] Koning Boudewijnstichting. *Projectoproep 2024: Visuele handicap & Inclusie – Voor meer toegankelijke vrijetijdsactiviteiten voor personen met een visuele beperking*. Tech. rep. Projectoproep ter bevordering van inclusieve vrijetijdsactiviteiten voor mensen met een visuele beperking. Accessed: 7 October 2025. Brussel, België: Koning Boudewijnstichting, 2024. URL: https://media.kbs-frb.be/nl/media/11548/HandicapVisuel_Folder.
- [21] Christina C Lessi et al. “5G and beyond 5G technologies enabling industry 5.0: network applications for robotics”. In: *Procedia Computer Science* 232 (2024), pp. 675–687.
- [22] Yuanwei Liu et al. “Robotic communications for 5G and beyond: Challenges and research opportunities”. In: *IEEE Communications Magazine* 59.10 (2021), pp. 92–98.
- [23] Bratati Mandal, Pritha Singh, and Sanatan Ratna. “Real-Time Object Detection and Environmental Feedback: A Smart Navigation System for the Blind Using AI”. In: *2025 12th International Conference on Emerging Trends in Engineering Technology - Signal and Information Processing (ICETET - SIP)*. 2025, pp. 1–6. DOI: [10.1109/ICETETSIP64213.2025.11156811](https://doi.org/10.1109/ICETETSIP64213.2025.11156811).
- [24] Lei Mu et al. “A Fine-Grained end-to-end Latency Optimization Framework for Wireless Collaborative Inference”. In: *IEEE Internet of Things Journal* 11.4 (Feb. 2024), pp. 7010–7022. DOI: [10.1109/JIOT.2023.3341628](https://doi.org/10.1109/JIOT.2023.3341628). URL: <https://doi.org/10.1109/JIOT.2023.3341628>.
- [25] National Institute of Standards and Technology (NIST). *Announcing the Advanced Encryption Standard (AES)*. FIPS PUB 197. 2001. URL: <https://csrc.nist.gov/publications/detail/fips/197/final>.
- [26] Waqas Nawaz, Kifayat Ullah Khan, and Khalid Bashir. “A Review on Path Selection and Navigation Approaches Towards an Assisted Mobility of Visually Impaired People.” In: *KSI Transactions on Internet & Information Systems* 14.8 (2020).
- [27] NETGEAR Inc. *Nighthawk M6 5G WiFi 6 Mobile Router (MR6110)*. <https://www.netgear.com/home/mobile-wifi/hotspots/mr6110/>. Accessed: 25 August 2025. 2025.
- [28] nPerf. *Baromètre des connexions Internet mobiles en Belgique (01/01/2024 – 31/12/2024)*. https://media.nperf.com/files/publications/BE/BE-Barometre-Mobile-connections-nPerf-2024_28012025.pdf. Accessed on September 4, 2026; published 07/02/2025. 2025.
- [29] nPerf. *Orange Mobile Network Coverage Map — Belgium*. <https://www.nperf.com/en/map/BE/-/201612.Orange-Mobile/signal?ll=50.88849724494421&lg=4.32938575744629&z=14>. Accessed on September 4, 2026. 2025.
- [30] Robert Odolinski et al. “Evaluation of the multi-GNSS, dual-frequency RTK positioning performance for recent android smartphone models in a phone-to-phone setup”. In: *Proceedings of the 2024 International Technical Meeting of The Institute of Navigation*. 2024, pp. 42–53.
- [31] J. Postel. *Transmission Control Protocol*. RFC 793. Sept. 1981. DOI: [10.17487/RFC0793](https://datatracker.ietf.org/doc/html/rfc793). URL: <https://datatracker.ietf.org/doc/html/rfc793>.
- [32] Kamran H. Sedighi et al. “Autonomous Local Path Planning for a Mobile Robot Using a Genetic Algorithm”. In: *Proceedings of the 2004 Congress on Evolutionary Computation (CEC 2004)*. IEEE, 2004, pp. 1338–1345. DOI: [10.1109/CEC.2004.1331039](https://doi.org/10.1109/CEC.2004.1331039).
- [33] Qianfeng Shen, Jun Zheng, and Paul Chow. “RIFL: a reliable link layer network protocol for data center communication”. In: *Journal of Optical Communications and Networking* 14.3 (2022), pp. 111–126.
- [34] Nikolai Smolyanskiy and Mar Gonzalez-Franco. “Stereoscopic First Person View System for Drone Navigation”. In: *Frontiers in Robotics and AI* 4 (2017). Open Access, p. 11. DOI: [10.3389/frobt.2017.00011](https://doi.org/10.3389/frobt.2017.00011). URL: <https://www.frontiersin.org/articles/10.3389/frobt.2017.00011/full>.
- [35] Spectrum-Tracker.com. *Belgium: 5G NR, 4G LTE, 3G WCDMA and 2G GSM Frequency Spectrum Bands*. <https://www.spectrum-tracker.com/Belgium>. Accessed on September 4, 2026. 2024.
- [36] Test Aankoop. *Interactieve kaart van het mobiel netwerkbereik in België*. <https://www.test-aankoop.be/hightech/gsms-en-smartphones/module/kaart-van-mobiel-netwerkbereik>. Geraadpleegd op September 4, 2026. 2025.
- [37] Web Real-Time Communications Working Group. *WebRTC 1.0: Real-Time Communication Between Browsers*. W3C Recommendation. Initial author: Ian Hickson; editors include Adam Bergkvist, Daniel Burnett, Cullen Jennings, Anant Narayanan, Bernard Aboba, Taylor Brandstetter, Jan-Ivar Bruaroey :contentReference[oaicite:1]index=1. Mar. 2025. URL: <https://www.w3.org/TR/webrtc/>.