TED University - Software Engineering

Multidisciplinary Engineering Analysis Report

Berkay Kaan KARACA-68317070956

Ceyda KUŞÇUOĞLU-16348076072

Kıvanç TERZİOĞLU-27233564574

1. Introduction

VAVI is a multidisciplinary engineering project designed to assist visually impaired individuals in indoor environments by providing safe, accurate, and real-time navigation support. The system combines indoor mapping, sensor fusion, computer vision, artificial intelligence, mobile application development, and human-centered design principles. Unlike traditional navigation systems that rely on GPS, VAVI operates entirely indoors using a fusion of Wi-Fi fingerprinting, inertial measurement units (IMU), camera-based object detection, and audio feedback.

This report presents a multidisciplinary engineering analysis of VAVI, focusing on how different engineering domains are integrated into a cohesive system. Additionally, a newly introduced feature—**AI-based visual localization using a single photo**—is analyzed as an extension beyond the originally documented system capabilities.

2. Problem Definition and Engineering Challenges

Indoor navigation for visually impaired users remains a challenging problem due to the absence of reliable GPS signals indoors and the high variability of wireless and sensor-based data. During the early stages of VAVI, Wi-Fi fingerprinting combined with node-based data collection was explored as a primary localization strategy. In this approach, signal strengths were collected at predefined graph nodes and used to train a localization model.

However, experimental results showed that Wi-Fi fingerprints were highly unstable across time, devices, and environmental conditions, leading to unacceptable localization errors. Due to these limitations, continuous navigation based on Wi-Fi and IMU fusion was abandoned as a core functionality.

The revised problem definition focuses on **robust initial localization** rather than continuous tracking. The current system determines the user's indoor position using **AI-based visual localization from a single photo**, after which it computes the **shortest path to a selected destination** using the A* algorithm. This design significantly improves reliability while reducing system complexity and sensor noise dependency.

3. Multidisciplinary Engineering Domains Involved

3.1 Computer Engineering & Software Engineering

The core architecture of VAVI is designed using modular software engineering principles. The system consists of a Flutter-based mobile application, a FastAPI backend, and multiple AI inference pipelines. Clean architecture, RESTful communication, version control, and automated testing practices are applied to ensure maintainability and scalability.

3.2 Artificial Intelligence & Machine Learning

VAVI heavily relies on AI techniques in two major areas:

- **Object Detection:** YOLO-based deep learning models (YOLOv5n / YOLOv8n) are used to detect obstacles such as people, doors, stairs, and walls in real time using the phone camera.
- **AI-based Visual Localization (New Feature):** A newly added capability allows the system to estimate the user's indoor location using a single photo. This is achieved by extracting visual features from the image and matching them against a pre-built indoor visual map using deep feature embeddings. This method enables rapid re-localization when Wi-Fi or IMU data is unreliable.

3.3 Signal Processing & Sensor Fusion

In earlier prototypes, VAVI experimented with Wi-Fi fingerprinting and node-based sensor fusion to achieve continuous indoor localization. Signal strength vectors collected at predefined nodes were used to train a machine learning model for position estimation. Despite extensive tuning, the approach suffered from environmental sensitivity and poor generalization.

As a result, sensor fusion is no longer used for continuous navigation. Instead, the system adopts a simplified and more robust pipeline where **AI-based visual localization** provides the initial position. Sensor data is retained only for orientation awareness and user interaction support, not for localization inference.

3.4 Mobile Systems & Embedded Computing

The system is optimized to run on standard Android smartphones without additional hardware. Engineering trade-offs are made to balance accuracy, latency, and battery consumption. Lightweight AI models and on-device inference (TFLite) are used to meet real-time performance constraints.

3.5 Human–Computer Interaction (HCI) & Accessibility Engineering

VAVI is designed with accessibility as a primary requirement rather than an afterthought. Visual interfaces are minimized, and the system communicates primarily through directional audio feedback. Stereo sound panning and intensity-based cues are engineered to convey distance and direction intuitively, ensuring safe navigation for visually impaired users.

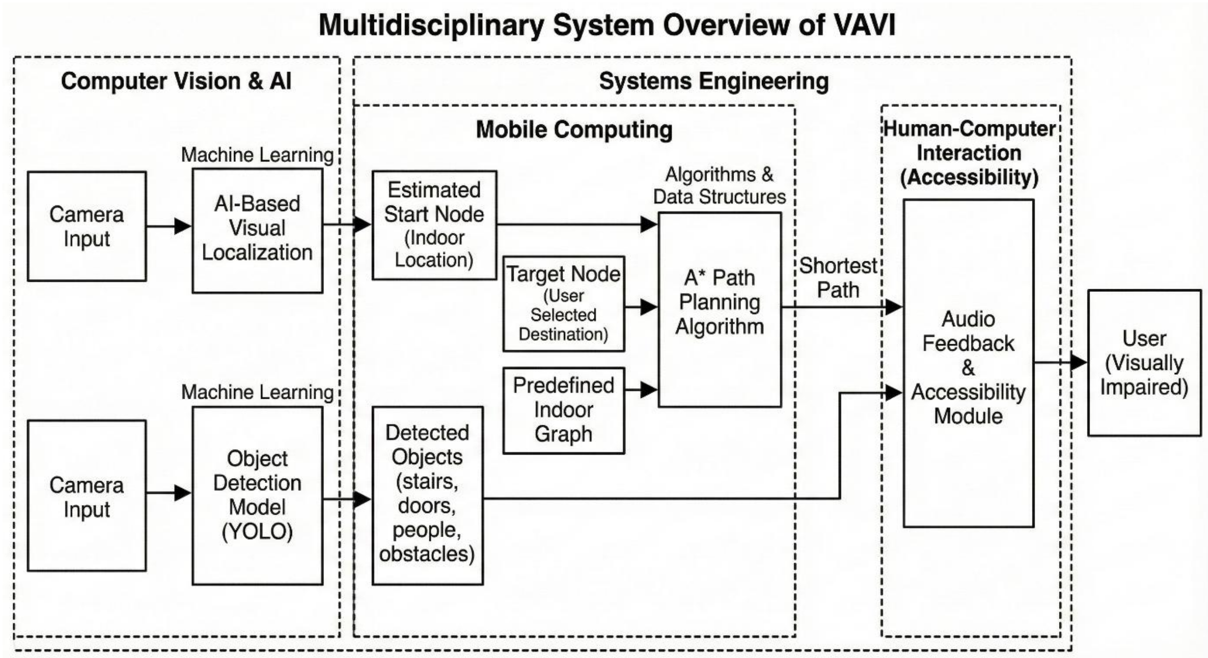
3.6 Systems Engineering

From a systems engineering perspective, VAVI integrates multiple subsystems—localization, perception, navigation, and user interaction—into a unified pipeline. Dependency management, fault tolerance, and fallback mechanisms (e.g., switching to visual localization when Wi-Fi degrades) are key design considerations.

4. System Architecture Analysis

The current architecture of VAVI reflects a design shift from continuous indoor navigation to **single-shot localization followed by optimal path computation**. The system operates in three main stages:

1. **Visual Localization Stage:** The user captures a photo of the surrounding environment. A deep learning model extracts visual features and matches them against a pre-built indoor visual map to estimate the user's current node.
2. **Path Planning Stage:** Once the starting node is determined, the system computes the shortest path to the selected destination using the **A*** search algorithm on a predefined indoor graph.
3. **Guidance Stage:** The resulting path is presented to the user as a clear and minimal route representation, without continuous re-localization.



This architecture reduces runtime uncertainty, avoids unstable wireless signals, and improves system determinism—an important property for safety-critical assistive systems.

5. Analysis of the New Feature: AI-Based Visual Localization

The newly added photo-based localization feature significantly enhances system robustness. Engineering analysis shows that:

- It reduces dependency on environmental Wi-Fi stability.
- It provides fast re-localization after user displacement or sensor failure.
- It introduces additional computational load, which is mitigated through optimized feature extraction and selective activation.

This feature demonstrates interdisciplinary integration of computer vision, machine learning, and systems engineering, extending VAVI beyond conventional indoor navigation solutions.

6. Risk and Reliability Analysis

Key risks include incorrect localization leading to unsafe guidance, slow AI inference, and sensor failures. These risks are mitigated through:

- Redundant localization sources
- Conservative navigation logic
- Continuous performance monitoring and testing

Safety-critical decisions prioritize obstacle avoidance over optimal path length.

7. Ethical and Social Considerations

VAVI addresses social inclusion by enabling visually impaired individuals to navigate complex indoor spaces independently. Ethical considerations include data privacy (no permanent image storage), transparency of AI decisions, and minimizing cognitive load on users.

8. Conclusion

VAVI is a strong example of multidisciplinary engineering in practice, combining software engineering, artificial intelligence, signal processing, mobile systems, and accessibility-focused design. The addition of AI-based visual localization further strengthens the system's reliability and innovation level. Overall, VAVI demonstrates how integrated engineering approaches can solve real-world, safety-critical problems and deliver meaningful social impact.

9. References

- IEEE Std 829-2008 – Software and System Test Documentation
- ISO/IEC/IEEE 29119 – Software Testing Standards
- Ultralytics YOLO Documentation
- Flutter & Android Sensor API Documentation