UNIVERSITY OF APPLIED SCIENCES
ASCHAFFENBURG

Software Requirements Specification

Project: SPOTY

Phase: Requirements Specification

Authors:
Zübeyir Eser, Mehmet Ali Özcevik, Mohamed Elsobky, Isam Hamid, Velat Perver Irmak

Documentname: AB-SPOTY-SRS-1

Version: 1.1

Creation date: 14.04.2024

File: https://github.com/ZuebeyirEser/Lidar

Modifications – Document Status

Version	Status	Creation Date	Editor	Modifications
1.0	Planned	14.04.2024	Zübeyir Eser	Initial Document
1.1	Under construction	24.04.2024	Zübeyir Eser	Several fixies in the SRS

(Status ::= planned, under construction, presented, accepted)

TABLE OF CONTENTS

1 Introduction

1.1 Purpose

The primary purpose of this program is to accurately detect the presence, location, and type of vehicles in a given environment. This information can be leveraged for various applications, including:

- **Autonomous vehicles:** Enabling self-driving cars to perceive their surroundings and navigate safely.
- **Traffic monitoring:** Providing real-time data on traffic flow and congestion for improved management.
- **Advanced driver-assistance systems (ADAS):** Supporting features like collision avoidance and lane departure warning.
- **Security and surveillance:** Detecting unauthorized vehicles in restricted areas.

1.2 Scope

This program focuses on vehicle detection using LIDAR data. The scope encompasses:

- Identifying various vehicle (e.g., cars, trucks, motorcycles).
- Determining the position and orientation of vehicles within the environment.
- Providing an output format compatible with downstream applications.

1.3 Definition, Acronyms and Abbreviations

LiDAR	Light Detection and Ranging
ADAS	Advanced Driver-Assistance Systems
Point Cloud	A collection of data points representing the three-dimensional coordinates of objects within the LIDAR's field of view.
SPOTY	Our product name.

1.4 References

Infrastructure, Object Detection Vehicle. (n.d.). [PDF file]. Retrieved April 24, 2024, from https://moodle.th-ab.de/pluginfile.php/379257/mod_folder/content/0/Projects/ObjectDetectionVehicleInfrastructure.pdf?forcedownload=1

OPEN3D] https://www.open3d.org/

1.5 Overview

The SRS will specify in detail the software requirements for the product SPOTY. Section 2 presents a general description of the SPOTY and it’s relationship within the operating environment (computer and external peripherals and systems) . A complete list and

description of the product functions and features will be provided. The type of user and user characteristics will be discussed. This Section also provides a discussion of any general constraints imposed on the product and any assumptions that are made regarding the operating environment of the product.

Section 3 will detail the software requirements of the product SPOTY. The behavioral requirements of the product SPOTY and operating environment will be discussed. The external interface, the hardware interfaces, the software interfaces, and the communication interfaces of the product will be outlined. Performance requirements will be discussed in section 3. Included in this discussion are operational requirements, exception handling, and testing requirements. Design constraints concern for the design phase of the product development will be addressed. Section 3 also describes the following product attributes: availability, security, maintainability, transferability and portability.

2 General Description

This section describes the general factors that will influence the design and requirements of our LIDAR-based vehicle detection program. It provides context for the detailed requirements outlined in later sections.

2.1 Product Perspective

(1) This program is an independent and self-contained software product. It is not designed as a component of a larger system.

(2) External Interfaces: The program will receive LIDAR data as input and provide vehicle detection information as output.

(3) Hardware Overview: The program is designed to run on a standard computer platform with sufficient processing power and memory to handle LIDAR data processing.

2.2 Product Functions

The primary function of this program is to utilize LIDAR data to:

- Detect the presence of vehicles, pedestrians, cyclists and e-scooter within its field of view.
- Determine the position and orientation of each detected vehicle.
- Provide output data containing this information in a specified format.

2.3 User Characteristics

The target users of this program are individuals with technical expertise in areas like robotics, autonomous vehicles, or related fields. They will have a strong understanding of LIDAR technology and its applications.

2.4 General Constraints

- **Regulatory Policies:** The program should comply with any relevant regulations regarding data privacy and security, especially if it is intended for deployment in real-world scenarios.
- **Hardware Limitations:** The program's performance will be limited by the processing power and memory of the chosen hardware platform. These limitations will be considered when defining performance requirements.
- **Criticality of the Application:** Depending on the intended use case (e.g., research project vs. safety-critical application), the program may have varying levels of criticality requirements.

2.5 Assumptions and Dependencies

- The program assumes the availability of a calibrated and functional LIDAR sensor that provides accurate 3D point cloud data.

- The program's functionality depends on the chosen LIDAR data format and communication protocol. Specific details will be outlined in the interface section.
- The success of vehicle detection relies on the quality and resolution of the LIDAR data provided.

3 Specific Requirements

This section details the specific functionalities, performance expectations, and attributes of the LIDAR-based vehicle detection program.

3.1 Functional Requirements

This subsection outlines the program's functionalities:

3.1.1 Vehicle Detection

- **Purpose:** Detect the presence of vehicles within the LIDAR data's field of view.
- **Inputs:** LIDAR point cloud data containing 3D coordinates of surrounding objects.
- **Operations:**
  - Apply filtering and segmentation algorithms to identify potential vehicle point clusters.
  - Utilize classification algorithms to differentiate vehicles from other objects.
- **Outputs:** List of detected vehicles with their corresponding 3D bounding boxes.

3.1.2 Vehicle Classification

- **Purpose:** Classify the type of detected vehicle (car, truck, motorcycle, pedestrians, cyclists, e-scooters).
- **Inputs:** Point cloud data of a detected vehicle.
- **Operations:** Analyze the vehicle's point cloud features (e.g., size, shape) for classification.
- **Outputs:** Classification label for the detected vehicle type.

3.1.3 Vehicle Localization

- **Purpose:** Determine the position and orientation of each detected vehicle.
- **Inputs:** Point cloud data and bounding box of a detected vehicle.
- **Operations:** Calculate cuboid in the point cloud (3D data).
- **Outputs:** Location (x, y, z coordinates) and orientation (roll, pitch and yaw) of the detected vehicle.

3.2 External Interface Requirements

3.2.1 User Interfaces

In our project we are going to use already exist UI which is OPEN3D.

3.2.2 Hardware Interfaces

The program shall be designed to communicate with a LIDAR sensor that meets the following specifications:

Using Lidar dataset from live observation of the data and using this dataset to identify all spespified vehicles.

3.2.3 Software Interfaces

This project does not require any external software products or interfaces with other application systems.

3.2.4 Communication Interfaces

The system will require a communication interface to receive data from the LIDAR sensor.
Interface: Wired at the corner of TH Aschaffenburg.
Data format: cuboid in the point cloud (3D data).

3.3 Performance Requirements

The vehicle detection system shall meet the following performance requirements:

3.3.1 Detection Accuracy

- **Vehicle presence detection:** The system shall detect the presence of vehicles with an accuracy of at least 95%.
- **Vehicle location accuracy:** The system shall determine the location of detected vehicles with an accuracy of at least 5 meters (horizontal) and 0.5 meters (vertical).
- **Vehicle type classification accuracy:** The system shall classify the type of detected vehicles (e.g., car, truck, motorcycle) with an accuracy of at least 85%.

3.3.2 Processing Speed

- **Real-time processing:** The system shall process sensor data and generate results in real-time, with a latency of no more than 50 milliseconds.
- **Scalability:** The system shall be able to handle data from multiple LIDAR sensors without significantly impacting processing speed.

3.3.3 Environmental Robustness

- **Lighting conditions:** The system shall operate effectively in various lighting conditions, including daylight, low light, and nighttime.
- **Weather conditions:** The system shall be able to operate in adverse weather conditions, such as rain, snow, and fog.

3.4 Design Constraints

3.4.1 Standards Compliance

The system shall comply with relevant industry standards and regulations, including:

- **Functional safety standards:** The system shall meet following functional safety standards to ensure safe operation.
    - IEC 61496-1:2020 LINK
    - ISO 13849-1:2023 LINK

- **Data privacy standards:** The system shall comply with data privacy regulations to protect user information.
    - GDPR LINK

3.4.2 Hardware Limitations

The system shall be designed to be compatible with the available hardware resources, including:

- **Computational power:** The system shall operate efficiently on the available computing hardware.
- **Memory constraints:** The system shall minimize memory usage to ensure efficient operation.

.

3.5 Software Systems Attributes

3.5.1 Maintainability

The system shall be designed to be easily maintainable, including:

- **Modular design:** The system shall be modular to allow for easy modification and updates.
- **Well-documented code:** The system code shall be well-documented to facilitate understanding and maintenance.

3.5.2 Reliability

The system shall be designed to be reliable and minimize downtime, including:

- **Error handling:** The system shall have robust error handling mechanisms to recover from unexpected situations.
- **Testing:** The system shall be thoroughly tested to identify and fix potential bugs.