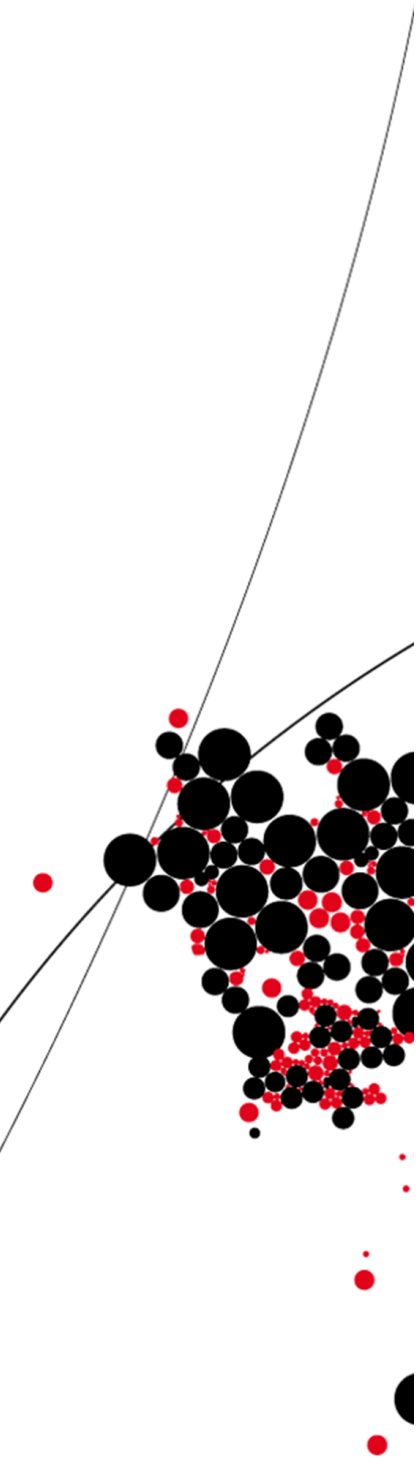
UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering,
Mathematics & Computer Science

Detecting and interfacing urban litter using computer vision.

Vincent P.G. Diks
B.Sc. Thesis
July 2022



Supervisors:
dr. F. Ahmed

Critical observer:
N. Bouali MSc

Creative Technology
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

Municipalities are continually innovating to limit litter and bulky waste deposition on public streets. Several efforts are made to make street clear ups more efficient and effective: collective litter registration platforms are developed by third parties, sweeper routes are optimized based on substantial parameters, and cloud-based artificial intelligence systems for trash detection are dawning.

This research aims at the development of a system that is capable of detecting urban litter using computer vision, and both the hardware and software specifications are explored. The system diagram that is developed in this research, should act as a blueprint to modern on-edge computer vision applications.

Using the Creative Technology Design Process and CRISP-DM data mining methodology, design choices and limitations are explored that contribute to the specifications. A data structure is developed to facilitate on-edge litter detection and the methodologies and techniques to interface the data are discussed. Data on litter occurrences have been modeled according to the essence of their use-case, grouped to road sections. The system has been developed to serve both users (contractors, maintainers and managers at the municipality) as well as external software to optimize sweeper routes. The viability of the system has been proven and considerations and advice are reflected upon to allow future development and improvement.

Acknowledgements

Foremost, I would like to express my special thanks of gratitude to my supervisor dr. Faizan Ahmed and critical observer Nacir Bouali for their guidance and academic advice that carried me through the project. Their profound understanding and competence in the field of computer science has contributed to the project as well as our collaboration. I am also grateful for their willingness to support my graduation project, which allowed me to pursue my interests and curiosities.

Next I would like to thank Foundit software and the municipality of Utrecht for providing me a thought-provoking and engaging research, and their eagerness to contribute to the project.

Last, I would like to thank the team at Enjoycleaningup for advertising and reminding me and people around the world of the significance of cleaning (urban) litter.

Contents

Chapter 1

Introduction

Litter is a global concern and affects a wide variety of ecological, social, and environmental systems. Litter development in urban areas is a result of deficient waste management and comprises not adequately disposed (e.g., misplaced trash) and contained waste (e.g., bulky waste able to scatter by wind or runoffs) [1]. Besides the effects of litter in urban areas, litter can transport from urban- to land areas or water bodies, consequently affecting non-urban areas.

The environmental consequences of macro- and micro anthropogenic debris are widespread, making total impact assessments unpractical with current methods. Nonetheless, quantifications of debris per area of interest are readily available, making various effects and responses researchable. Litter, in its total variety of magnitudes, and plastics are associated with effects on wildlife regarding bioaccumulation, reproduction, and toxicity, as well as smothering and entanglement [2]. Among others, urban and coastal landscapes are reportedly affected by litter. Besides the aesthetic impacts, potential economic implications arise to both maritime and continental activities, such as fishery, aqua- and agriculture and tourism.

Municipalities are continually innovating to limit litter and bulky waste deposition on public streets. Several efforts are made to make street clear ups more efficient and effective: collective litter registration platforms are developed by third parties, sweeper routes are optimized based on substantial parameters, and cloud-based artificial intelligence systems for trash detection are dawning.

To enable such innovations, the interaction between maintainers and the situation in the physical world shifts to a digital environment [3]. Presuming readiness for Industry4.0 as described by Ghobakhloo, a digital environment should shadow the state of litter within the area to be maintained. This premise requires the development of new methods to visualise litter on maps, together with the underlying techniques to quantify litter so that it can be used in digital systems.

This paper investigates methods to contribute to the optimization of sweeper routes in urban environments with the use of image recognition systems mounted on municipality-owned vehicles.

1.1 Research questions

The main research question for this bachelor thesis is derived from client requirements and is formulated as follows: *How can a fully fledged system be designed to accommodate autonomous urban litter recognition with valuable data interfaces?* The definition of *valuable* is defined with respect to the client requirements and its fulfilment by the system.

To answer this research question, its subquestions have been divided into three categories, accompanied with their respective subquestions:

1. What physical system can allow accurate live object detection on moving vehicles, considering scalability, practicality, and industrial integration?
    (a) What hardware specification is required to facilitate object detection on a moving vehicle?
    (b) What object recognition framework provides optimal recognition abilities, based on accuracy, precision, and performance of the detection?
    (c) How can the hardware be setup to provide low maintenance and future developments?
2. How can a data model be developed for automated litter recognition?
    (a) What object classification software can provide optimal reliability and inference speed to allow detection and classification of litter?
    (b) How can representative training data be collected and maintained to facilitate future development?
3. How can trash detection best be visualized on an online platform to improve sweeper routes and make better use of existing cleaning resources?
    (a) What is the optimal data-structure to provide fast and reliable storage of detected trash items and export functionalities for route-optimization software?
    (b) How can detected litter be represented on a map to reflect street cleanliness indices?
    (c) What interface can allow resource planners to optimize sweeper routes?

1.2 Structure of the report

The report is hereinafter structured as follows. In Chapter 2, the State of the Art in technology for each three subtopics as defined in the research questions is discussed. This should provide an overview of existing methods and technologies and serves as a starting point for the development of the system. In Chapter 3, the methods, techniques and structure of the research are discussed. Chapter 4 explores the development process of the system, with intermediate results and reflections to the research questions. Chapter 5 concludes the findings in Chapter 4 and summarizes the key takeaways. Chapter 6 discusses limitations with regards to the research and terms several aspects for future work.

Chapter 2

State of the Art

Designing a interdisciplinary system capable of proper object detection and having meaningful interaction with municipalities requires in-depth knowledge on the state of the art in related technologies. In order to carry out a comprehensive analysis of related works and review associated concepts, four publication databases (Google Scholar, Scopus, ResearchGate, Papers with code) were benefited from. Some literature found on the internet through generic search engines is also reviewed for their contribution to related topics. This research intends to investigate state-of-the-art technologies that share functionalities or ambitions with this research.

First, the concepts for manual assessment and parameterisation of street cleanliness are discussed to introduce considerations and connections to the current methodology. In addition, methods for visualising the measured data are discussed. Second, the characterization of AI processors for object detection workloads are discussed. Lastly, frameworks and methodologies for the detection of litter are discussed, as well as available data for development. This way, all related disciplines are touched upon to identify pitfalls and opportunities to assist the ideation process of the prototype. NB: these three disciplines do not imply an existing combination in technology.

2.1 Street cleanliness assessment

2.1.1 Public space quality assessment in municipalities

This section investigates the methodology used to assess the quality of public space advised to municipalities. Practical tasks for the removal of litter are the primary focus, grounded by policy making.

The platform for knowledge exchange in the fields of infrastructure, transport and traffic and public space, known as CROW, provides a national standard for quality assessments of public space in the Netherlands. This standard allows public space managers, in general municipalities, to communicate clearly with their respective managers, users, and maintainers [4]. Besides the facilitation of meaningful communication, the standard provides parameterisation of quality in respect to area per time, supported by visual scale bars [4].

The application of this standard is divided into 4 components.

1. Ambition: policy choices in terms of ambition themes.
2. Assignment: formulation in terms of performance requirements.
3. Supervision: supervision of whether performance requirements have been met.
4. Monitoring: review whether ambition levels have been met.

2.1.2 Determination of measuring areas for manual measurements

Measuring areas describe a specific stretch of public space to accommodate a score for a specific assessment category (e.g. bulky litter). These areas are predetermined according to, not mutually exclusive, three methods. These methods are established from the following premises [4].

1. All measuring areas combined should at least cover 90% of the area to be maintained by contractors.
2. All measuring areas should cover at least 0.05 hectare and at most 1 hectare, with a length of at most 200 meters.

Grid method The grid method overlays the area to maintain with a grid of cells spanning at most $100 \times 100meters$. Each cell represents an area. In general, this method does not provide a one-to-one representation of the area to be maintained and lacks sufficient variety in objects to provide a veracious score.

Logic border methods The logic border method requires a detailed map that outlines logic borders, like: building boundaries around squares or the boundaries of vegetation. An area is defined around these logic borders. Each defined area should be of roughly equal size.

Selective location method The selective location method defines areas by picking locations representing the average public image of the maintained area. The location is demarcated by a radius of at most 50 meters.

2.1.3 Visualization of focus areas and litter densities

Focus areas are considered areas where resources require the most attention, either in a direct or as soon as possible manner [4]. Prioritization methods are of importance to establish a meaningful, beneficial, and pragmatic visualization, such that municipalities become empowered in supervising and monitoring the work and orchestration of contractors out in the field.

Most standards agree, to some extent, on the representation of street cleanliness within bounded areas, and accordingly assess the areas with an ordinal score. Each higher level area, that is, an area consisting of multiple measuring areas, is given the score based on the worst case area that is entailed [4]. With this method, areas that require immediate attention are assured to be communicated with managers, users and maintainers.

The ordinal classification of a specific audit area can be represented in a choropleth map type, modelling the spatial distribution of litter opposed to areas (as polygons) with set bounds. Choropleth maps assume the homogeneity of their areas [5]. With the advance of constant sampling methods, unaggregated data points lead to the continuous representation of the data through the means of isopleth maps, now widely adopted [6].

The introduction of isopleth maps for the representation of litter allows the user to select locations of interest and attain an expected value. When using geostatistics, the need for data collection on that specific location becomes avoidable, assuming data saturation in a significantly close perimeter. Spatial interpolation algorithms are used to compute the continuous representation of data. Semivariogram inference and the kriging model are two algorithms that are adapted industry wide [7]. However, these two techniques assume stationarity of data locations, which does not apply to the heterogeneous measuring areas with abrupt changes in litter data. [7] proposes a method to overcome the requirement for stationarity by using map data to estimate the local mean of the random function, to which kriging can be applied. However, this method requires a constant local mean, which is not the case for litter as can be seen in the cleanliness assessment of [8].

[9] presents a geostatistical method to represent both areal and point data within spatial interpolation of continuous trash properties. This method relies on area-to-point kriging, which is

> "used to map the variability within geographical units while ensuring the coherence of the prediction so that the sum or average of disaggregated estimates is equal to the original areal datum. The resulting estimates are then used as local means in residual kriging." [9]

2.2 Hardware requirements for edge computing

Edge computing is a computing paradigm that locates computations at the location of the data source. This method induces several advantages opposed to cloud computing solutions, like: low latency, high privacy, more robustness, and more efficient use of network bandwidth [10]. The majority of edge AI processors available to the industrial market, are designed for very specific deep learning operations [11]. This entails the trade-off between performance, accuracy and cost, and introduces an offset in expected performance between generic computer benchmarks (like memory and clock speed) and designs of AI processors [11]. [11] proposes three metrics to evaluate and compare edge AI processors: *accuracy, latency, and energy efficiency*.

[11] compares the performance of three popular AI processors using the open-source data set named *MS COCO*, which provides 100 object classes of common items, running in YOLOv2, the object detection system [12]; this configuration mimics the application of litter detection [13].

Specification	Edge TPU	Nvidia Xavier	NovuTensor
Terra -operations per second (TOPS)	4	22.6	15
Memory	32-bit LPDDR4	256-bit LPDDR4X	128-bit DDR4
Power (watt)	2.5	15	20

These three processors yield no significant difference in accuracy [11]. In terms of latency, however, Nvidia's Xavier performs **5.28X faster** than Edge TPU, and NovuTensor performs **3.8X faster** than Edge TPU. These results strongly indicate that $operations/s \propto 1/latency$. No stable proportionality becomes evident from the difference in energy efficiency of the processors. Edge TPU delivers **1.13X higher** energy efficiency then Nvidia's Xavier, and **1.04X higher** then NovuTensor [11].

Inference time is one of the main characteristic that defines the processing power of the required hardware. The quality of the detection is defined by the object detection system software, as will be elaborated upon in the *Object detection* section.

2.3 Object detection

Object detection is the detection of class instances in an image, reported with a form of pose information like: location, scale or in terms of a bounding box [14]. [14] suggests that object detection methods fall into two parts, *generative* and *discriminative*.

Generative object detection employs a probability model, oftentimes structured as neural network, for the pose variability of objects in respect to their background.

Discriminative object detection employs a parametric model for posterior probabilities, which derives its values from a set of training data.

In other words, a generative model focuses on explaining how the data was generated, while a discriminative model focuses on predicting the labels of the data.

2.3.1 Object detectors

For the case of discriminative models, there are three main types of object detectors: two-stage, single-stage and transformer based detectors [15]. A network that utilizes a distinctive module for the proposal of objects is termed two-stage. These detectors conduct propositions for a maximum amount of objects within an image before classifying and localizing the objects. Single-stage detectors, on the other hand, directly predict image pixels as objects using dense sampling. This main feature of single-stage detectors benefits from using predefined boxes and keypoints to locate objects [15]. Two-stage detectors show greater accuracy and precision, however lack global context, have higher inference times, and require complex structures as opposed to two-stage detectors [15].

Chapter 3

Methods and Techniques

In this chapter, the methodologies used for the design of the system and proposals are discussed. First, a swift overview is given of the Creative Technology Design Process. Second, a more software-centered design approach – CRISP-DM – is discussed. Both methodologies have been combined to fit and facilitate the design of this project.

3.1 Creative Technology Design Process

In figure 3.1 the four-step design process is displayed within the discipline of Creative Technology [16]. In the initial phase termed *Ideation*, the probable and non-probable solutions to the problem statement are described. By coming up with various options, the solution inherits a form of creativity [16]. For this project, however, there are set specifications and well-established output metrics, limiting the variety in approaches to the solution. During the second phase, known as the *Specification*, project requirements are formulated based on knowledge acquired during the first phase [16]. These requirements are fully determined by the CRISP-DM method in the results section, since those exhibit the multi-disciplinary scope in both the software, business and ecological domain. The third phase, the *Realisation*, explores the requirements to create a working model of the solution. Ultimately, a working prototype is created along which validation is conducted [16]. *Validation* is the fourth phase during the process, in which the system is parametrically tested on its target audience and/or users [16].

Although this process is taken as a starting point during the development of the system, a more extensive and industry-renowned methodology is used to complement and streamline the design.

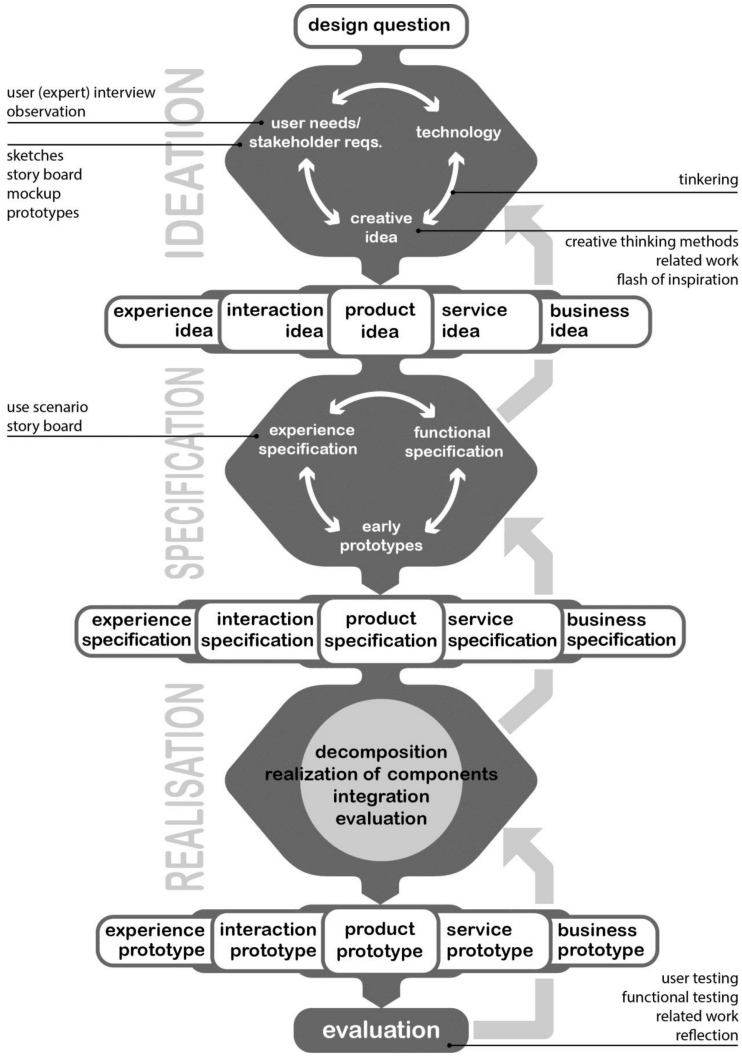


Figure 3.1: Diagram of the Design Process for Creative Technology [16]

3.2 CRISP-DM

CRISP-DM stands for *cross-industry standard process for data mining*, and is a open standard commonly used for data mining purposes. In figure 3.2 a diagram is shown depicting the iterative process. This methodology entails six steps to facilitate a data mining project from start to finish, and is described as follows by [17]:

1. *Business Understanding* gives an overview of available and required resources and establishes the goal to the data mining project.
2. *Data Understanding* focuses on the exploration of data, as well as the statistical analysis on the quality of data.
3. *Data Preparation* is a model dependent preparation step in which new attributes are generated and insufficient data removed based on the initial requirements.
4. *Modeling* consists out of the selecting and building the model, along with its test case.
5. *Evaluation* is the phase in which the results of the model are checked against the defined business objectives. Next to that, the process is reviewed in general.
6. *Deployment* consists of planning the deployment, monitoring and maintenance and does so in a descriptive manner.

3.3 Hybridization of methods

The Creative Technology Design Process does not fully expedite and direct the steps needed to develop this project, whereas CRISP-DM does not cover non-data related challenges. Since this project is divided into three main topics: *hardware design, computer vision software design, and information output software design*, both models are be applied to their own specificity with respect to the research questions. Some concepts from CRISP-DM are reapplied to design processes beyond data mining.

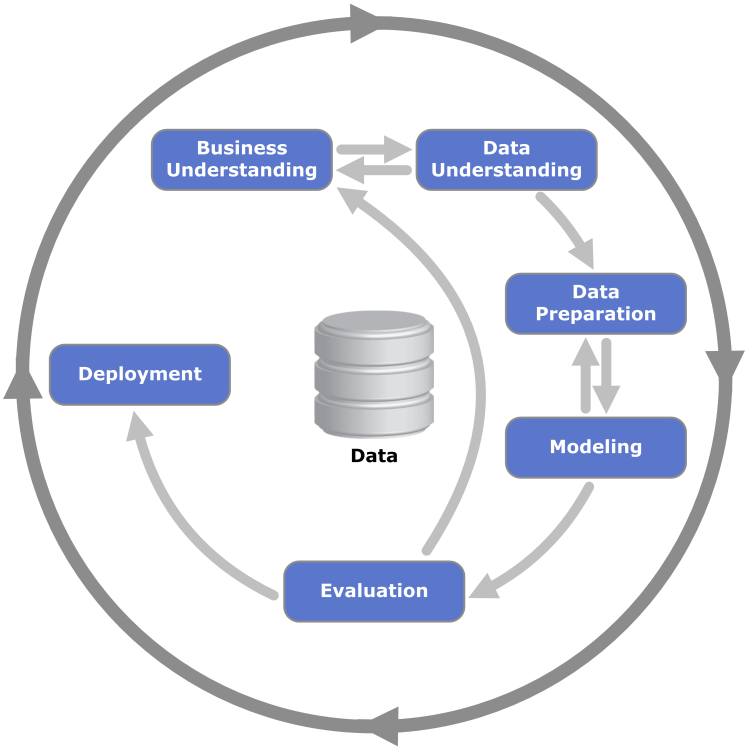


Figure 3.2: Diagram of the CRISP-DM [18]

Chapter 4

Development of system

4.1 Hardware design

For the development of the physical system, key parts from CRISP-DM are utilized. The business understanding and the statistical analysis of compute power mainly substantiate the design choices. The system validation is conducted according to empirical research.

4.1.1 Business understanding

A mobile computer vision system that is operated by municipalities is likely to be offered through tenders [19, pp. 51–55]. Therefore, the cost-efficiency should account for a two-year runtime of the system. Due to the mobility of the system, the system is likely to either locate its image analysis on a server, or on the device that captures the image stream [20]. A third option is a hybrid of the two [21], although this option does not affect the design choices based on business specification.

Server computation allows low-effort hardware scalability and full control over buffering in the processing of the images. A large drawback to server computation is the need for a continuous data stream from the devices to a central server, which is needed to perform immediate analysis. Data prices in 2021 in the Netherlands hovered around € 3,- per gigabyte for consumers [22], whereas industrial data ranges up to €8,- per gigabyte based on the top 5 google search results. When sweeper cars drive for 8 hours per day, 20 days per month, outputting a single 1080p H.264 encoded video at 25FPS (9 gigabytes per hour), at a price in the range of € 5,- per gigabyte. Each vehicle will cost an additional € 7.200,- per month to transfer data to a central server.

Edge-computing runs the inference model on the device, yielding advantages such as high privacy, reduced data transmission rate, and reduced latency [23]. This approach

Figure 4.1: rolling shutter, motion blur, and frequency of capture

requires each vehicle to be equipped with a dedicated computer capable of handling the inference at the required speed. Data costs for standard industrial appliances like these (not abusing the network) range up to € 100,- per month.

Given the fact that top-tier on-edge devices sell for up to € 3000,- per device, it's clear that on-edge computation is more cost-effective than server computation for this specific use-case.

4.1.2 Data understanding

The use of this system is on the one hand to directly respond to high occurrences of trash, and on the other to gain information on the efficiency of street sweepers. For the latter use-case, data should be collected prior to cleaning the area, suggesting a front mounted camera to capture the street. Attaching a camera to a moving vehicle brings several challenges along, o.a.: *rolling shutter, motion blur, and frequency of capture*. In Figure 4.1 the effects of these challenges are shown and how they result in a lack of data. Rolling shutter distorts the shape of objects due to the the serial reading of camera sensors. Motion blur drags color data over several pixels of the image sensor, resulting in blurry images. Capturing pauses can lead into deficits in the patchwork of the images. These challenges are all affected by the shutter speed and capture speed.

4.1.3 Ideation

In general, an on-edge system follows the architecture depicted in Figure 4.2. During the ideation phase, multiple approaches and solutions to each specific element in the architecture are tested to both determine and fulfill the specifications of the system.

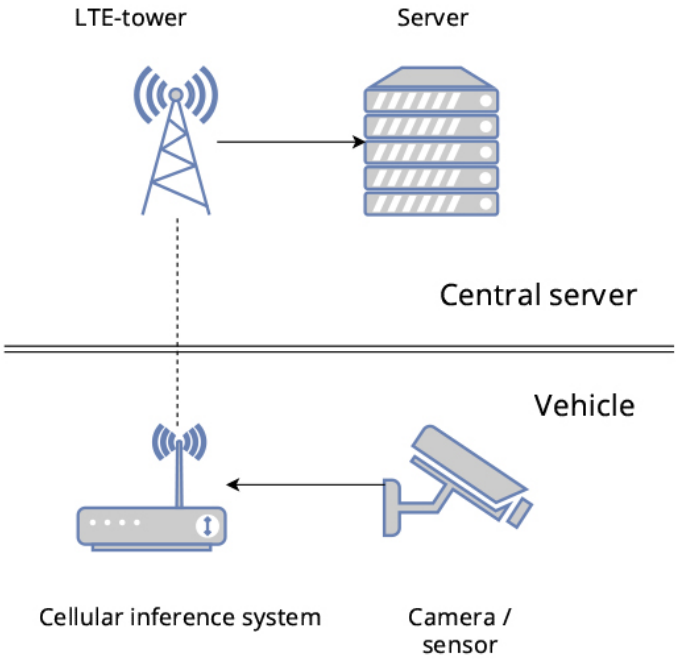


Figure 4.2: Basic hardware architecture on-edge inference system

Camera

In order to determine the position and requirements of the camera, as well as the effects of a moving vehicle, a test setup (see A.1 and A.2) has been developed with the camera specifications in Table A.2. The Raspberry Pi HQ camera is a non-production camera that allows direct sensor control through a serial (CSI) interface [24]. Due to this full sensor control, several parameters have been altered to test its influence on the image quality.

Camera mount The camera mount is tested in two position mounted on top of a 1.60m high car, facing backwards (in line with the longitudinal axis of a car) and facing sideways (in line with the lateral axis of a car). See A.2 for the setup. In Figure 4.3, the difference between longitudinal and lateral images is shown. As a result of the forward momentum (over the longitudinal axis), objects in the longitudinal mounted camera setup traverse over near pixels during the sensor (digital) opening and reading, whereas the lateral setup causes objects to be captured over multiple horizontal pixels. This effect results in the horizontal motion blur seen in the right image of Figure 4.3.

Shutter speed The shutter speed of a camera is essentially the time that a sensor allows its pixels to count photons, and affects the brightness and motion blur of the image. Since the used camera is not a DSLR, meaning there is no physical element controlling the shutter speed, some digital operations have to be performed to count the amount of photons in

Figure 4.3: Longitudinal and lateral image output.

a pixel on a constantly illuminated sensor. The sensor is designed such that the pixels are arranged in a matrix layout, consisting out of rows and columns, and operations are performed on row level [25]. The two operations available to this sensor are *read* and *reset*, respectively counting the amount of photons on a pixel and removing the count stored to that pixel. The time of illumination is the difference in time between the occurrences of the *reset* function on the same row, thus not dependent on the reading speed of the image. Speeds of 3ms to 100ms were tested. At a minimum, speeds of 20ms are required in bright situations (sunny sky at noon) to provide sufficient granularity.

As a side effect of the digital shutter, the *rolling shutter effect* is introduced. This effect causes radial warping of objects and occurs when objects move over the pixel plane during the opening of the digital shutter.

From the empirical tests, it became evident that the camera has to be mounted along the longitudinal axis of the moving vehicle and handle a minimal sensor reading speed. This will yield the highest granularity in image quality.

Industrial camera The Raspberry Pi HQ camera is susceptible to damaging high peak voltages due to its direct connection between the power input and processor of the host machine [24]. To prevent the risk of hard bricking and to make the camera resistant to outdoor effects, the AIDA outdoor POV camera is used for further testing and proposed in the eventual system. This camera is IP67 rated and provides an IP data interface with roughly the same image quality. However, full sensor control is not available on this model.

Cellular inference system

Deep learning models resource usage is intensive due to the amount of operations required to run an inference model. The inference system is required to handle the minimally required OPS (operations per second) to run real time data analysis. As will be seen in the section *Computer vision software design*, non-optimized systems can run YOLOv5 (the model used

Model	TX2	Xavier	Orin
OPS	$\approx 10.4TOPS(1.33TFOPS)$	25 TOPS	275 TOPS
Power max	15W	20W	60W
Price in 2022	€ 500	€ 1300	€ 2000
Year introduced	2014	2018	2022

Table 4.1: Technical comparison Jetson family as of 2022 [28]

in a later stage) at an estimated 1.2 TOPS/FPS (Terra OPS per frame per second). Assuming an image has to be taken every 1 meter, the following relation holds: *min tops = speed in ms / tops per fps*. This means that at 40 km/h, roughly 10 TOPS is needed to run the inference model at the required speed. For the use-case of sweepers, it can be assumed that speeds higher then 40 km/h will not occur.

The Neural Compute Stick 2 (NCS2) is a neural accelerator in USB form capable of handling 4 TOPS. The first prototype used this accelerator to run inference models on the Raspberry Pi 3B+ setup described in the *Camera ideation* subsection. However, due to the non-integrated nature of the system (USB connection to host), the inference speed did not exceed 2FPS, whereas 4.8 FPS was expected.

Although a variety of technology companies worldwide produce and market their AI chips and systems, NVIDIA delivers and entire optimization ecosystem around it [26], [27]. NVIDIA's Edge AI family is named Jetson and offers multiple models like described in Table 4.1.

For this project, the TX model is used, since it fulfills the minimal OPS requirement and was directly available. Additionally, the TX2 model is the only device optimized for using 16 bit floating point values for the active models, against the 8 bit integer optimization used on the other devices. This comes in convenient with the 16 bit architecture of the neural network used in the software development. Using either of the Jetson family allow the same code base to be used over different devices, depending on the processing power needed. A fanless, Ubuntu based carrier board, provided by Advantech, is used to facilitate the NVIDIA board.

4.1.4 Evaluation

It has become evident that there is enough ecosystems and devices available on the market to develop this system. Due to the increasing demand, there is a variety of research groups and companies working on developing new and optimized hardware, therefore it is advised

Model	TACO	Open Litter Map	Litter	Domestic Trash Dataset
Available images	3600	>100k	14k	>9k
Classes	28	11	28	10
Subclasses	60	178	3	-

Table 4.2: Top 4 litter datasets [30]

to use the state-of-the-art and accommodate software development for a longer period of time then with hardware available to the masses.

4.2 On-edge software design: computer vision

4.2.1 Business understanding

There are two parts to the software design of the total system, on the one hand there's the server software capable of putting litter occurrences to use, on the other hand, there is the software ran on the edge for object detection (in other words: server and edge system). The computer vision software design aims at implementing the logic capable of accurately detecting objects within the peripheral range of a vehicle. The on-edge system must be capable of outputting a data representation of litter occurrences, classifications and positions. Furthermore, the system utilities should maintain data buffers to ensure system continuation in case of low data connectivity.

4.2.2 Data understanding

Given the multiple classes that define litter (e.g. cigarettes, cups, cardboard, etc.), the required inference speed and the priority of counting occurrences over yielding accuracy in intra-class discrimination, the object detecting model will be developed according to a one-stage convolutional neural network detector.

In Figure A.3, the top 5 popular object detection algorithms are shown along with the inference speed in ms. Figure A.4 displays the mean Average Precision (mAP) based on the COCO dataset of these algorithms. mAP defines how precise a model is able to recall objects, and is computed by comparing the overlap between guessed and annotated bounding boxes over the objects in the dataset. This project will be developed according to the YOLO model, due to its predicted high inference speed and accuracy [29].

To train the model, a dataset with annotated data is necessary. Table 4.2 displays the top 4 available datasets based on quality, quantity and licensing.

Figure 4.4: Semantic segmentation and bounding box classification

Based on the cost and versatility of the models in Table 4.2, the TACO-dataset is selected to train the model on.

4.2.3 Data preparation

YOLO is a one-stage bounding-box image detector [29]. The annotated data (labels) in the TACO-dataset is, contrary to YOLO, instance segmented, as shown in Figure 4.4. So in order to train the model in Darknet – an open source neural network framework written in C and CUDA [31] – the labels have to be converted from polygons to box coordinates with width and height attributes. With a conversion script written in Python, the dataset was prepared for Darknet. The conversion script worked by selecting the min and max value on both the x and y axis of the polygon that describes the segmentation. By subtracting these values the proportions were yielded, and by taking the average of these values the centroid of the box was yielded.

The model has been trained on model v5 of YOLO, which shares the features of v4, but is written according to the Pytorch framework [32]. This simplifies the training process significantly.

To reliably validate the model, the TACO-dataset has been split into a train, validation, and test set according to a 80/10/10 ratio. Although impossible to distinguish these hyper-parameters a-priori [33], the optimal advantage can be taken from enlarging the train set. This is due the dataset lying in the short bound of the convergence range of 4.000 objects for Darknet [31]. So instead of taking a commonly used ratio of 70/20/10, a more result centered approach has been taken.

4.2.4 Modeling

YOLOv5 is pretrained on COCO, meaning it inherits detecting features of common objects. The training of the dataset yielded the results in Figure 4.5. The model is trained over the v5m (medium) version of the network, which is a 16 bit floating point network, 41MB in size. The model was exported in ONNX (Open Neural Network Exchange, an open standard for machine learning interoperability.) format, to allow TensorRT optimization in the NVIDIA environment. TensorRT [34] is a NVIDIA native SDK that optimizes inference speeds using layer fusion. The engine creates the execution context of a model, corresponding to a CUDA stream. This allows for parallelization of inference execution on a GPU [35].

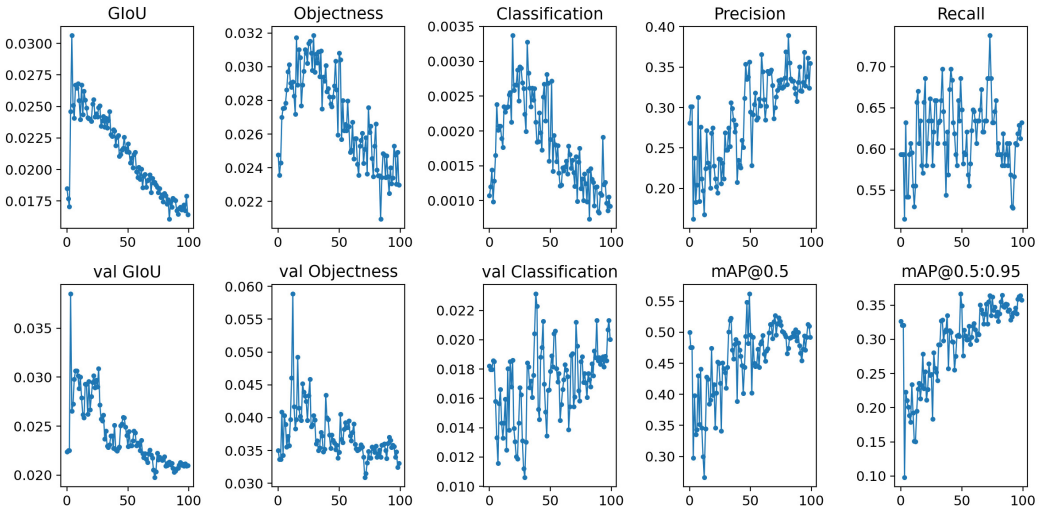


Figure 4.5: Parametric results of YOLOv5s train with TACO-dataset

4.2.5 Evaluation

Several significant results are evaluated based on the specifications of the model.

- **Generalized Intersection over Union** (GIoU). Intersection over Union is a metric to evaluate how close a prediction bounding box is to ground truth. This loss function is not differentiable though, making it unsuitable for backpropagation. The generalized version, comprising of a value between -1 and 1, with 0 being exact overlap, is able to provide a differentiable loss function [36]. After full training, the GIoU has converged to zero, indicating a high capability in object localization.

- **Precision and Recall**. Like stated in *Business understanding*, the use-case of the model attaches more value to the recognition of trash rather than its intra-class discrimination. With a recall of 0.63 (stating true positives over true positives plus

false negatives), the dataset exceeds expectations of the v5m model [12], but can not be considered fully accurate. This value has to be taken into account during estimation of street litter occurrences in the server software development. However, when comparing litter occurrences within a specific area, the probability of not detecting trash is equal over each street. So when prioritizing streets based on litter frequencies, this inaccuracy is normalized.

It has been shown that there are multiple object classification software available that can provide high or optimal reliability in relation to the inference speed. Furthermore, representative training data is readily available, however, could be enhanced in case of a larger available budget.

4.3 Server software design: interfacing for resource optimization

4.3.1 Business understanding

The server implementation, and the connection to the server, should provide a system to represent litter data and output it to both human users (contractors, maintainers), as well as external optimization software. Furthermore, user output should consider the conjoinment of the system with the existing, manual assessments as described in the section on public space quality assessment.

4.3.2 Realisation

A system diagram of both software systems (on-edge and server) is displayed in Figure 4.6. Each part component will be discussed with relevant design choices.

Server

The central server to which on-edge devices can be connected is a dedicated VPS with 8GB RAM running Windows IIS 2019. This setup has no particular advantage nor disadvantage over NGINX or Apache at this scale, but was selected due to its immediate availability.

Instead of implementing an API to transfer data from the devices to the server, the system uses native PostgreSQL replication. Replication is the automated process of copying data from various *slaves* (in this case on-edge device) to a single or various *masters* (server). Data consistency is perpetuated even in the asynchronous mode, where slaves lag behind the master. Replication also has built in partition tolerance, which recovers arbitrary data

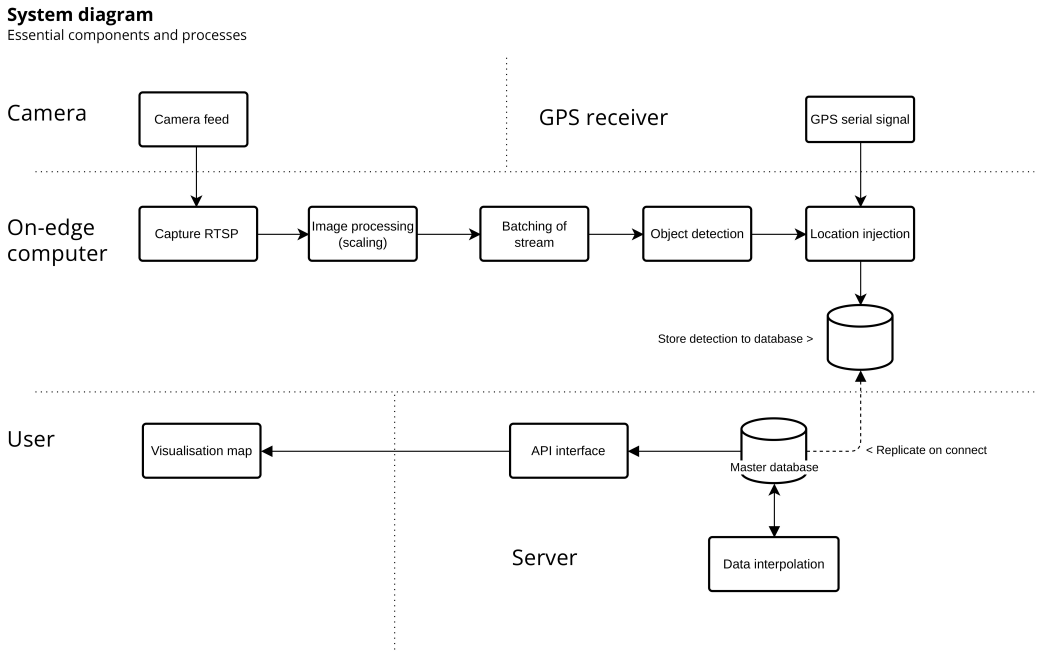


Figure 4.6: System diagram of software over all physical systems

messages lost along transfer. The replication has been setup in a single-master configuration, meaning the slaves are not aware of data collected on other slaves [37]. Since only data has to be sent to the server, and no data structures, only physical replication is used. This means that the system will move data *as is* (binary) to the server.

This process of replication is facilitated through the opening of ports on both the server and on-edge devices, and maintained during connection. The test area (municipality of Utrecht) has full 5G and 4G coverage, so no large buffers should be created.

Database design

To accommodate the collected trash position, the database scheme has been developed according to Figure 4.7.

Point The point class holds information about the location of litter or images. PostgreSQL support spatial classes, so in production, lat-lng and x-y are of type *Point*. Lat-lng is the polar representation, whereas x-y is the location on the Cartesian plane with unit in meters. The usage of the Cartesian values serves two purposes. First, to compute the shortest path between images with specific anomalies. Second, to assign an object to a road segment. More on the latter in the section on *spatial functions and interpolation*.

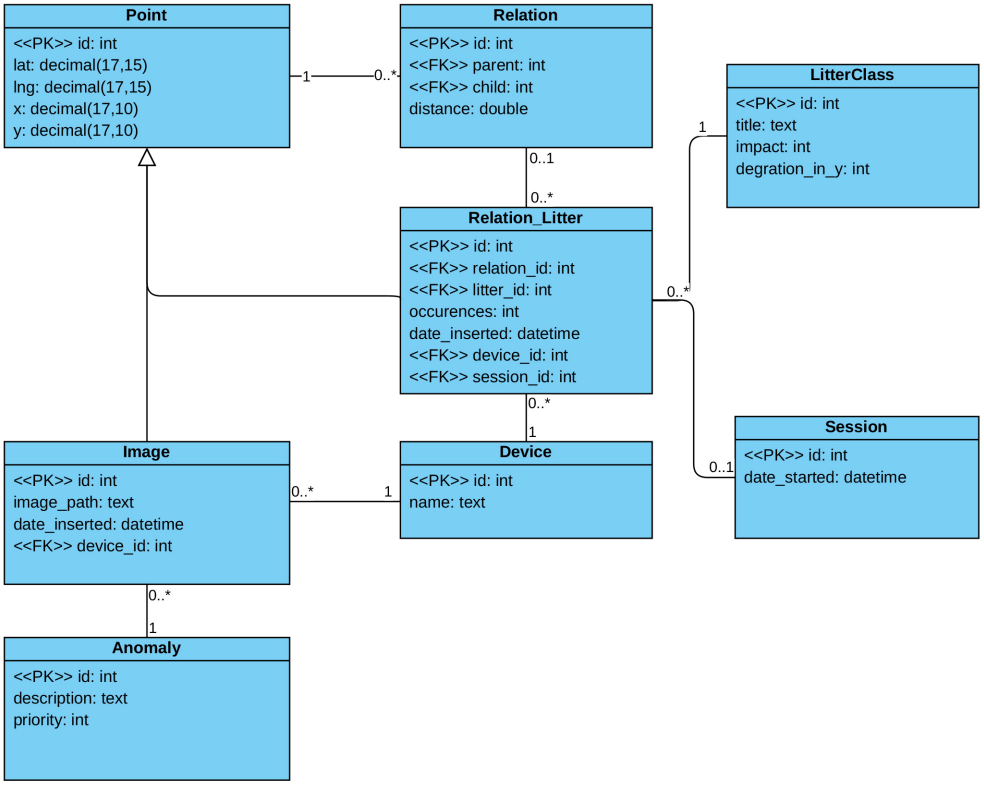


Figure 4.7: Database design, replicated over server and on-edge devices.

Relation Relations represent (sections of) roads, and can be compared to segments of geoJSON polygons in modern map software that define road structures.

Relation Litter This class assigns a location to the litter class (which are represented by the categories from the TACO-dataset). Furthermore, the device that captured the litter is stored and as well as to what road segment the location belongs.

Session To segregate different moments of measurements, a session is started each time a new road segment (relation) is entered by the vehicle. This way, each time a segment is driven over, data becomes available about a single and the last drive by. This prevents multiple detection of the same litter and allows for insights over time.

Image The image class stores information about images taken on the device. This serves two purposes, first to gain new training data, second to give contextual information about anomalies on the road. N.B: images are not taken at every moment, and if taken, not necessarily transmitted over LTE directly. Images are not stored as binary, but as JPEG's in the file system of the device. The WebDAV-protocol [38] is used for file transmission of the image files.

Spatial functions and interpolation

The first approach to displaying litter occurrences to the users was based on the idea of interpolating litter frequencies. However, the mutually exclusive presence and scattering effects of litter objects disallowed this approach. A more meaningful approach was taken, and served both the user output as well as the parametric output to external route optimization software. Since sweepers can be assumed to finish a road segment once they start cleaning it, litter frequencies can be bound to the road segments.

To assign litter to a specific road segment, given that the road segment consists of a starting coordinate and an ending coordinate, the function in Listing 4.1 was implemented in PostgreSQL to return the closest segment.

Listing 4.1: SQL function to find closest road segment, point class JOIN statements left out on purpose

```
set @radius = 150 #meters
set @point_x = ?
set @point_y = ?
SELECT
    ST_Distance(
        point(@point_x, @point_y),
        LINESTRING(r.from_coordinate, r.to_coordinate)
        ) as distance_to_r
FROM relation r
WHERE r.x + @radius > @point_x #optimization
AND r.x - @radius < @point_x
AND r.y + @radius > @point_y
AND r.y - @radius < @point_y
LIMIT 1;
```

With this database structure, litter can be grouped to relations given the last session value. This will return a list of road segments with distance per segment and a count of litter, from which a litter per meter value can be obtained.

This data is all available through a PHP 7.4 based API. This API has a role-based authentication (using Codeigniter 3 with Ion Auth) system that protects its endpoints, allowing a versatile purpose. On the one hand, the data is used to draw the current pollution situation using geoJSON's represented with segment data, on the other hand, the data can be accessed by other software in a JSON format.

4.3.3 Evaluation

Reflecting on the research question, it has become evident a highly efficient data-structure can be developed using PostreSQL, and comes with reliable protocols to replicate data from the on-edge devices to the server. Furthermore, litter can be represented grouped to the corresponding road segment, as it will provide enough granularity for users and optimization software. Last, the API-interface provides data for both external software as well as custom front-end systems.

Chapter 5

Conclusion

The viability of this system is highly dependent on each individual and diversified research topic. Based on the evaluations in the development of the system, it can be concluded that the complete system is viable. Various techniques have been explored and showed the considerations and design choices that had to be made to allow for interoperability of the subsystems while maintaining the output quality of that specific subsystem.

It has been shown that an accurate and efficient object detector can be developed with existing datasets. Furthermore, this research has shown that hardware is available on the market to accommodate this and future software systems. However, vehicle-adjusted camera's require use-case considerations and ultimately software enhancements to adjust for their mobile conditions. Last, litter can be represented in relation to road segments to allow proper data representation that is meaningful for both users and external software.

5.1 Notes for development

Since this paper serves to develop the technicalities of the described system, as well as advises future development, some statements are required in conclusion rather than future work.

The field of on-edge computer vision – and AI – is rapidly progressing and advancing. This means that focus on improving general software and algorithms is highly likely to go concurrent with others improving the same solutions. For the purpose of using the proposed system, independency of subsystems are a prerequisite to the development.

Chapter 6

Limitations and Future Work

This sections explores the peculiarities of the research and limitations with respect to the research. Limitations on system level are incorporated in the research. Furthermore, advice is given for research on the same topics.

This research consists of multidisciplinary research topics. This often caused interoperability challenges that drew attention away from the main research. Although the system has been fully developed on both hardware and software side, no evaluation has been conducted with the fully installed system. This was mainly caused by hard bricking the hardware in the last phase of the research. Furthermore, not all hardware devices were available at the vendors, requiring to work with substitute devices for the majority of the research.

This research is focused on the technical aspects of the system development and the interaction users and external software undergoes. Note that video recording public space, people, residential areas and processing this data using AI has many legal implications under both Dutch and EU law.

Future work is needed to test the system in practice and compare the accuracy with the theoretical statistics.

Bibliography

[1] L. Lebreton and A. Andrady, "Future scenarios of global plastic waste generation and disposal," *Palgrave Communications*, vol. 5, no. 1, pp. 1–11, Dec. 2019. DOI: 10.1057/s41599-018-0212-7. [Online]. Available: https://ideas.repec.org/a/pal/palcom/v5y2019i1d10.1057_s41599-018-0212-7.html.

[2] W. W. Y. Lau, Y. Shiran, R. M. Bailey, *et al.*, "Evaluating scenarios toward zero plastic pollution," *Science*, vol. 369, no. 6510, pp. 1455–1461, 2020. DOI: 10.1126/science.aba9475. eprint: https://www.science.org/doi/pdf/10.1126/science.aba9475. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.aba9475.

[3] M. Ghobakhloo, "Industry 4.0, digitization, and opportunities for sustainability," *Journal of Cleaner Production*, vol. 252, p. 119 869, 2020, ISSN: 0959-6526. DOI: https://doi.org/10.1016/j.jclepro.2019.119869. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0959652619347390.

[4] CROW, *Kwaliteitscatalogus Openbare ruimte 2013: Standaardkwaliteitsniveaus voor Onderhoud*. CROW, 2013.

[5] W. Mu and D. Tong, "Mapping uncertain geographical attributes: Incorporating robustness into choropleth classification design," *International Journal of Geographical Information Science*, vol. 34, no. 11, pp. 2204–2224, 2020. DOI: 10.1080/13658816.2020.1726921. eprint: https://doi.org/10.1080/13658816.2020.1726921. [Online]. Available: https://doi.org/10.1080/13658816.2020.1726921.

[6] M. VOLTZ and R. WEBSTER, "A comparison of kriging, cubic splines and classification for predicting soil properties from sample information," *Journal of Soil Science*, vol. 41, no. 3, pp. 473–490, 1990. DOI: https://doi.org/10.1111/j.1365-2389.1990.tb00080.x. eprint: https://bsssjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1365-2389.1990.tb00080.x. [Online]. Available: https://bsssjournals.onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2389.1990.tb00080.x.

[7] T. Hengl, G. B. Heuvelink, and A. Stein, "A generic framework for spatial prediction of soil variables based on regression-kriging," *Geoderma*, vol. 120, no. 1-2, pp. 75–93, 2004.

[8] M. Zamorano, "An index to quantify street cleanliness," *Waste Management*, vol. 140, pp. 135–144, 2010.

[9] P. Goovaerts, "A coherent geostatistical approach for combining choropleth map and field data in the spatial interpolation of soil properties," *European Journal of Soil Science*, vol. 62, no. 3, pp. 371–380, 2011. DOI: https://doi.org/10.1111/j.1365-2389.2011.01368.x. eprint: https://bsssjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1365-2389.2011.01368.x. [Online]. Available: https://bsssjournals.onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2389.2011.01368.x.

[10] Y.-L. Lee, P.-K. Tsung, and M. Wu, "Techology trend of edge ai," in *2018 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, 2018, pp. 1–2. DOI: 10.1109/VLSI-DAT.2018.8373244.

[11] Y. Hui, J. Lien, and X. Lu, "Three-dimensional characterization on edge ai processors with object detection workloads," in *Int. Conf. for High Performance Computing, Networking, Storage, and Analysis*, 2019.

[12] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018.

[13] T.-Y. Lin, M. Maire, S. Belongie, *et al.*, *Microsoft coco: Common objects in context*, 2014. DOI: 10.48550/ARXIV.1405.0312. [Online]. Available: https://arxiv.org/abs/1405.0312.

[14] Y. Amit, P. Felzenszwalb, and R. Girshick, "Object detection," in *Computer Vision: A Reference Guide*. Cham: Springer International Publishing, 2020, pp. 1–9, ISBN: 978-3-030-03243-2. DOI: 10.1007/978-3-030-03243-2_660-1. [Online]. Available: https://doi.org/10.1007/978-3-030-03243-2_660-1.

[15] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee, "A survey of modern deep learning based object detection models," *Digital Signal Processing*, vol. 126, p. 103514, 2022, ISSN: 1051-2004. DOI: https://doi.org/10.1016/j.dsp.2022.103514. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1051200422001312.

[16] A. Mader and W. Eggink, "A design process for creative technology," Undefined, in *Proceedings of the 16th International conference on Engineering and Product Design, EPDE 2014*, E. Bohemia, A. Eger, W. Eggink, A. Kovacevic, B. Parkinson, and W. Wits,

Eds., ser. Eamp;PDE, null ; Conference date: 04-09-2014 Through 05-09-2014, The Design Society, Sep. 2014, pp. 568–573, ISBN: 978-1-904670-56-8.

[17] C. Schröer, F. Kruse, and J. M. Gómez, "A systematic literature review on applying crisp-dm process model," *Procedia Computer Science*, vol. 181, pp. 526–534, 2021, CENTERIS 2020 - International Conference on ENTERprise Information Systems / ProjMAN 2020 - International Conference on Project MANagement / HCist 2020 - International Conference on Health and Social Care Information Systems and Technologies 2020, CENTERIS/ProjMAN/HCist 2020, ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2021.01.199. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050921002416.

[18] IBM, *Crisp-dm help overview*. [Online]. Available: https://www.ibm.com/docs/en/spss-modeler/18.2.0?topic=dm-crisp-help-overview.

[19] M. A. B. Chao-Duivis and R. Kluitenberg, *Parlementaire geschiedenis Aanbestedingswet 2012*. Instituut voor Bouwrecht, 2013.

[20] C. Esposito, A. Castiglione, F. Pop, and K.-K. R. Choo, "Challenges of connecting edge and cloud computing: A security and forensic perspective," *IEEE Cloud Computing*, vol. 4, no. 2, pp. 13–17, 2017. DOI: 10.1109/MCC.2017.30.

[21] J. Guo, B. Song, S. Chen, F. R. Yu, X. Du, and M. Guizani, "Context-aware object detection for vehicular networks based on edge-cloud cooperation," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 5783–5791, 2020. DOI: 10.1109/JIOT.2019.2949633.

[22] Cable.co.uk., *Western europe: Mobile data price 2021*, Jul. 2021. [Online]. Available: https://www.statista.com/statistics/1123435/price-mobile-data-europe/.

[23] M. Rohith, A. Sunil, and Mohana, "Comparative analysis of edge computing and edge devices: Key technology in iot and computer vision applications," in *2021 International Conference on Recent Trends on Electronics, Information, Communication Technology (RTEICT)*, 2021, pp. 722–727. DOI: 10.1109/RTEICT52294.2021.9573996.

[24] M. P. Layer, T. T. Solutions, P. Lefkin, and R. Wietfeldt, "Understanding mipi alliance interface specifications,"

[25] *Camera hardware documentation*. [Online]. Available: https://picamera.readthedocs.io/en/release-1.13/fov.html.

[26] A. Elkaseer, M. Salama, H. Ali, and S. Scholz, "Approaches to a practical implementation of industry 4.0," *Resource*, vol. 3, p. 5, 2018.

[27] G. Pang, "The ai chip race," *IEEE Intelligent Systems*, vol. 37, no. 2, pp. 111–112, 2022. DOI: 10.1109/MIS.2022.3165668.

[28] *Jetson modules*, May 2022. [Online]. Available: https://developer.nvidia.com/embedded/jetson-modules.

[29] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[30] A. Mikołajczyk, *Agamiko/waste-datasets-review: List of image datasets with any kind of litter, garbage, waste and trash*. [Online]. Available: https://github.com/AgaMiko/waste-datasets-review.

[31] J. Redmon, *Darknet: Open source neural networks in c*, http://pjreddie.com/darknet/, 2013–2016.

[32] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, *Yolov4: Optimal speed and accuracy of object detection*, 2020. DOI: 10.48550/ARXIV.2004.10934. [Online]. Available: https://arxiv.org/abs/2004.10934.

[33] S. Oymak, M. Li, and M. Soltanolkotabi, "Generalization guarantees for neural architecture search with train-validation split," in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, Jul. 2021, pp. 8291–8301. [Online]. Available: https://proceedings.mlr.press/v139/oymak21a.html.

[34] NVIDIA, *Https://developer.nvidia.com/tensorrt*. [Online]. Available: https://developer.nvidia.com/tensorrt.

[35] E. Jeong, J. Kim, S. Tan, J. Lee, and S. Ha, "Deep learning inference parallelization on heterogeneous processors with tensorrt," *IEEE Embedded Systems Letters*, vol. 14, no. 1, pp. 15–18, 2022. DOI: 10.1109/LES.2021.3087707.

[36] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union," Jun. 2019.

[37] Z. Böszörményi and H.-J. Schönig, *PostgreSQL Replication*. Packt Publishing, 2013.

[38] E. J. Whitehead and Y. Y. Goland, "Webdav," in *ECSCW '99: Proceedings of the Sixth European Conference on Computer Supported Cooperative Work 12–16 September 1999, Copenhagen, Denmark*, S. Bødker, M. Kyng, and K. Schmidt, Eds. Dordrecht: Springer Netherlands, 1999, pp. 291–310, ISBN: 978-94-011-4441-4. DOI: 10.1007/978-94-011-4441-4_16. [Online]. Available: https://doi.org/10.1007/978-94-011-4441-4_16.

[39] R. Pi, *Buy a raspberry pi high quality camera*. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-high-quality-camera/.

[40] S. Valley, *Specifications aida hd-ndi-ip67*. [Online]. Available: https://www.streamingvalley.nl/product/aida-imaging-hd-ndi-ip67/.

[41] J. Tao, H. Wang, X. Zhang, X. Li, and H. Yang, "An object detection system based on yolo in traffic scene," in *2017 6th International Conference on Computer Science and Network Technology (ICCSNT)*, 2017, pp. 315–319. DOI: 10.1109/ICCSNT.2017.8343709.

[42] paperswithcode, *Object detection on coco test-dev*, 2022. [Online]. Available: https://paperswithcode.com/sota/object-detection-on-coco.

Appendix A

Development

Sensor	Sony IMX477R stacked, back-illuminated
Resolution	12.3 megapixels
Sensor diagonal	7.9 mm
Pixel size	1.55 µm $\times$ 1.55 µm
Output	RAW12/10/8, COMP8 over CSI

Table A.1: Raspberry Pi HQ camera specifications [39]

Sensor	1/3″ Progressive CMOS
Resolution	1080p, 1080i, 720p
Sensor size	5.346mm x 3.003mm
Pixel size	2.75 µm (H) x 2.75 µm (V)
Output	IP (NDI|HX/SRT/RTMP/RTSP)

Table A.2: AIDA POV camera specifications [40]

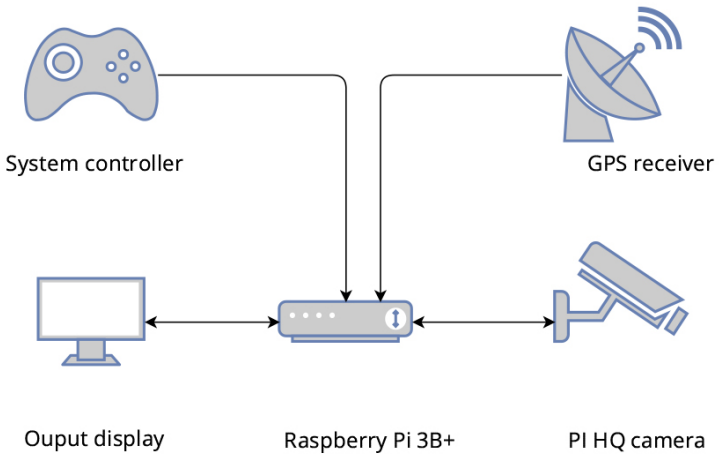


Figure A.1: Camera test setup

Figure A.2: Camera test physical setup. Inside, abreast and side mount.

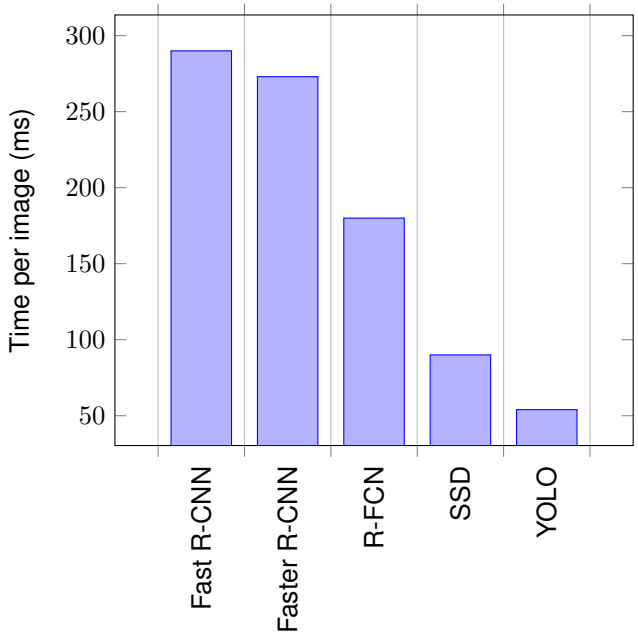


Figure A.3: Speeds of 5 popular detection models [41]

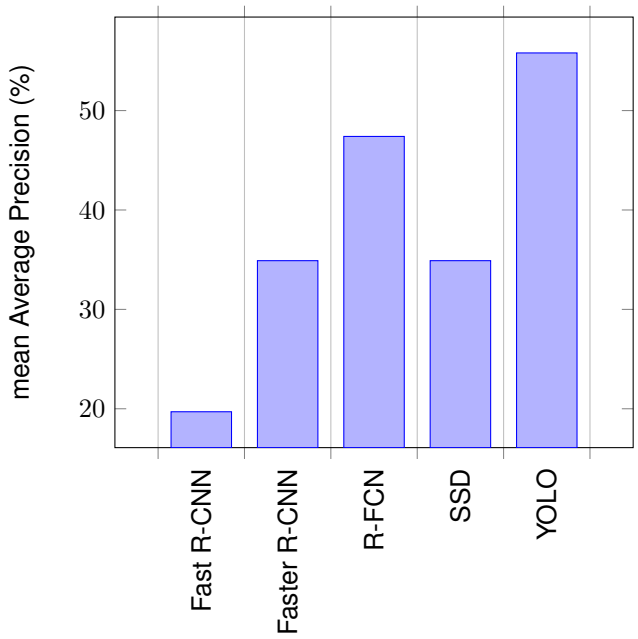


Figure A.4: mAP on COCO of 5 popular detection models [42]