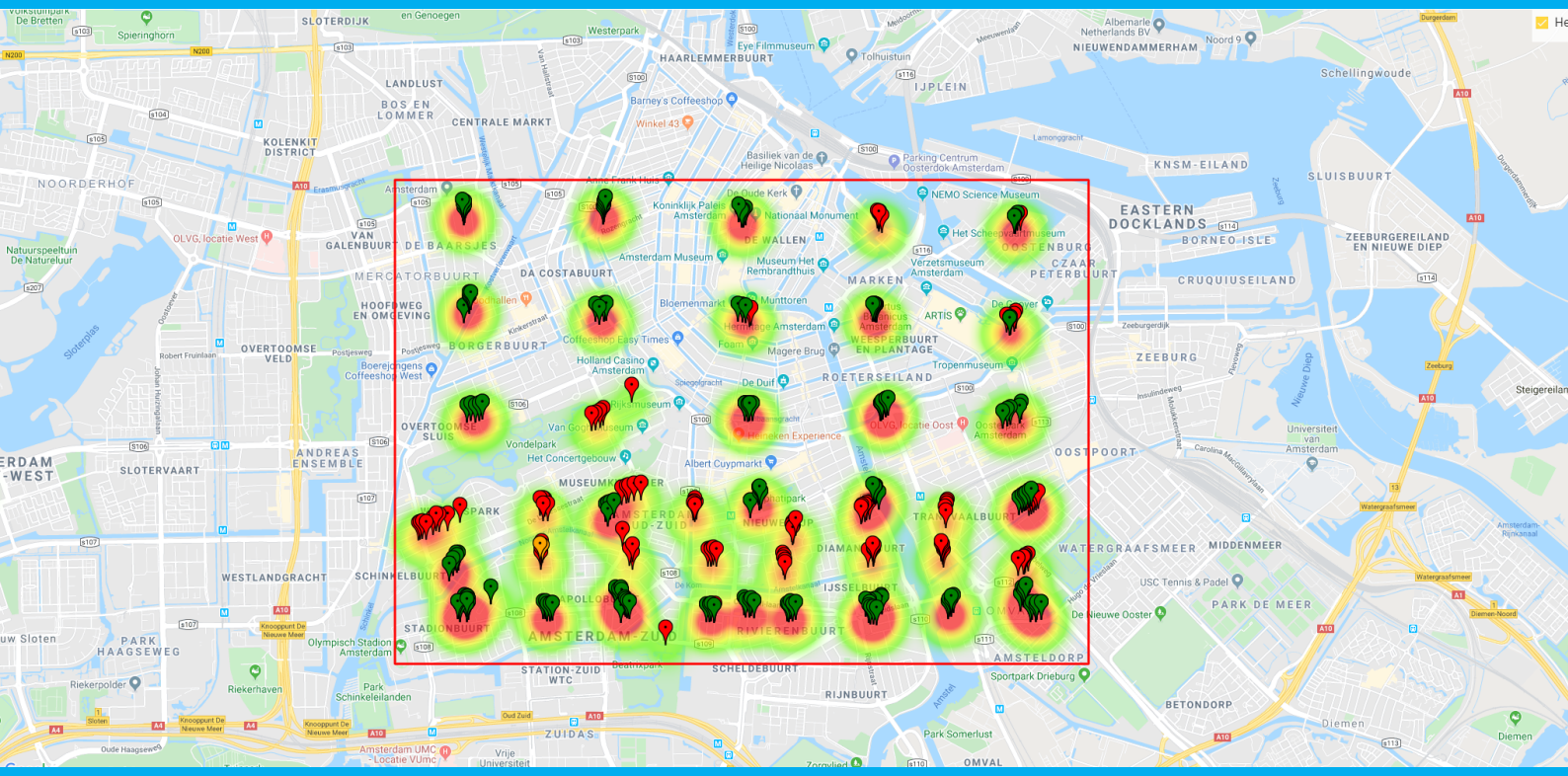
A multi-platform crowd-mapping application for urban object mapping using street-level imagery

Gerard van Alphen



TU Delft

A multi-platform crowd-mapping application for urban object mapping using street-level imagery

by

Gerard van Alphen

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday May 14, 2020 at 01:00 PM.

Student number:	4303512	
Project duration:	May 1, 2019 – May 14, 2020	
Thesis committee:	Prof. dr. ir. Alessandro Bozzon,	TU Delft, supervisor
	Dr. ir. Christoph Lofi,	TU Delft
	Dr. ir. Achilleas Psyllidis,	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Abstract

Crowd-mapping is a relatively new field of research, which involves the collection of geographic data by a crowd of workers. The collection of said data is of great importance for organizations like municipalities, where it is used for applications such as maintaining streets and greenery. The benefit of crowd-mapping over traditional mapping methods, where workers physically observe the area, is that it has the potential to be far more cost-effective and time-efficient. As this should not come at the cost of losing accuracy, research needs to be done on how to effectively map objects in a city.

Although previous work has focused on mapping urban objects using street-level imagery, they are all specifically aimed at a single type of object. Furthermore, they do not offer a general method for geo-location estimation and do not estimate the height of the objects. All of the systems designed in previous work only support task execution using a web platform. As crowd-mapping is nothing without the crowd, it is important to keep the workers engaged. No research had been done on how the task execution platform and type of task could affect the worker engagement and satisfaction.

In this thesis we will design a system for crowd-mapping urban objects using street-level imagery. We will propose novel methods for geo-location and object height estimation. Experimentation showed that the proposed geo-location method was able to deliver an accuracy with up to 83% of the estimations being within 2.5 meters of the ground truth with a mean distance of 1.85 meters. The height estimation showed up to 85% of the estimations being within 30 centimeters from the ground truth with a mean difference of 15 centimeters. Furthermore, the system supports task execution on three platforms; web, mobile and mobile virtual reality. We demonstrated the feasibility of executing mapping-, data-enrichment- and verification tasks on each of these platforms. Experimentation with the different platforms showed that the type of task and execution platform affects user engagement, cognitive load, satisfaction and execution time.

Preface

This thesis report contains my findings on the subject of *a multi-platform crowd-mapping application for urban object mapping using street-level imagery.* In this thesis I try to answer the question "*How can we design a multi-platform crowd-mapping application for urban object mapping using street-level imagery?*". Using the knowledge gained from previous work and expanding on this work, a design was composed and implemented. This resulted in an crowd-mapping system supporting task execution on mobile, web and mobile virtual reality. The feasibility of this product was evaluated and demonstrated during a experimentation period.

My initial interest for the subject of crowd-mapping was sparked when following the CS4145 Crowd Computing course given by Alessandro Bozzon and Nava Tintarev. As part of the project for this course, my team and I developed a proof-of-concept crowd-mapping application with gamification elements. When Alessandro made it evident that he had a spot available for further research into the subject I let him know I was interested and the rest is history.

During this project Sihang Qiu helped me out by providing feedback and answering all my questions almost instantly after asking them on Slack. He has been a tremendous help and for that I want to thank him. Furthermore I want to thank Alessandro Bozzon for the provided feedback and giving me the opportunity to do this thesis with him as a supervisor. Finally I would like to thank the thesis committee and the experiment participants for dedicating their time and helping me out in this project.

Gerard van Alphen
Delft, May 2020

Contents

1 Introduction

1.1. Motivation

An urban environment is full of objects like trees, trash bins and lamp posts. Gathering data such as geo-location and height of these objects is of great importance for planning and maintenance purposes for municipalities. For example, when conducting maintenance on greenery in a city, a dataset of the location and type of greenery can be used to send out workers and bring the right equipment. Traditionally mapping objects involves having municipal workers document each individual object by physically observing the environment, which is a labour intensive activity. This makes the traditional way of mapping the objects in an urban environment both time-consuming and costly, as cities cover a large surface which need to be explored by the paid workers.

Services like Google StreetView and Mapillary offer up-to-date 360 degree views of city environments, with high spatial coverage. Such services make street level imagery data easily available, which also allows researchers to access it and develop applications with this data. To help solve the problem of mapping urban objects street level imagery can be used, as it eliminates the need to physically be in the environment to do observations.

A time- and cost-efficient approach would be to use machine learning to recognize objects in images, which has been an active area of research, also for tasks such as cataloging urban trees [21, 22, 37]. Machine learning does have some drawbacks however, as it requires a large amount of good quality training data to have the potential to be accurate. Even when enough training data is available, machine learning might still fail to estimate the geo-location. The image quality may not be sufficient and as a result it might not recognize the entire object. This might happen for example when an object is obstructed or when it blends in with its background. In many cases human observers will easily recognize such scenarios.

To reduce the labour intensive nature of physically mapping urban objects and overcome the drawbacks of machine learning, micro-task crowd-sourcing offers a solution. Having a crowd of workers execute a simple online web-based task for a reward, like annotating objects, can provide data requesters with the required data.

Qiu et al.[27] have researched crowd-mapping urban objects using street-level imagery, in which crowd sourcing and street-level imagery are combining in a web based platform. Workers get the task of annotating a certain object by drawing a rectangle, after which the geo-location of said annotation is estimated. Other notable research was conducted by Saha et al. with Project Sidewalk [31, 32], in which they developed an application for auditing urban accessibility using street level imagery. This research shows that the combination of street-level imagery and crowd-sourcing is a viable way for mapping urban objects.

Several crowd-sourcing platforms currently exist, such as Figure Eight[1] and Amazon Mechanical Turk[2]. They offer a service for general crowd-sourcing purposes, but lack the specific requirements for crowd-mapping campaigns as this demands spatial task assignment and scheduling strategies. A general purpose crowd-mapping system using street-level imagery does not exist yet and the previous research such as the work by Qiu serves as a proof of concept and therefor this is a good starting point but further research is required.

[1] https://www.figure-eight.com/
[2] https://www.mturk.com/

The previous work focuses on mapping a single type of object or area such as trees or accessibility issues. Furthermore, they do not offer a general method for geo-location estimation which can be used without exploiting undocumented APIs or a service other than Google Street View. None of the systems estimate the height of the objects, which for example could be of use for maintenance of greenery in a city. These systems are all implemented for task execution in a web browser. Crowd-mapping as the name suggests is completely dependent on actually having a crowd of workers. Therefor it is important to have workers engaged and satisfied with the platform, as this might determine their willingness to continue executing the tasks and returning to the platform to execute more tasks. Project Sidewalk observed a high user dropoff, with only 29.5% of the users completing at least one task. They also mention they want to add smartphone support at a later stage to increase engagement[31, 32]. This study will check how different task execution platforms affect factors like output quality, execution time and worker satisfaction.

The output quality is of great importance for data requesters, as the data needs to be reliable. Similarly execution time will determine how quick they can access the data. The system will support task execution on web browsers, smartphones and mobile virtual reality. Previous work such as the study by Ma et al. [23] shows that executing crowd-sourcing tasks using virtual reality is feasible. Mobile virtual reality is a more affordable and therefore more accessible means to experience virtual reality when compared to the dedicated virtual reality devices (such as the Oculus Rift and HTC Vive), as it only requires a smartphone and a headset with lenses in which the phone can be strapped (such as Google Cardboard and Samsung Gear VR). Therefor more workers will be able to execute these tasks which makes it more useful at scale. Allowing task requesters to deploy tasks and analyze the results themselves, would make this a fully-fledged crowd-mapping system and additionally developing an application supporting multiple task execution platforms would allow for experimentation.

1.2. Challenges

In order to realize the aforementioned application, a number of challenges will need to be tackled. The first challenge relates to high-quality output of the tasks. The output of a crowd-mapping task should be a collection of objects with corresponding properties. The most important property is the geo-location, which needs to be estimated based on the annotation of the crowdworker and the geo-location of the panorama camera which created the street level image. For some tasks it may also be interesting to estimate the height of an elevated object, which is not trivial. Furthermore, a task requester might be interested in gathering text based information about the object, which should be intuitive and convenient to provide by the crowdworkers.

Another challenge lies in the design and implementation of the application. As it should support multiple platforms for task execution, the activity of crowd-mapping should be abstracted. Furthermore, the task should only have to be created once and will then be deployed across all the platforms. The task creation and analysis user interface should be intuitive to use.

Finally, experimentation will have to be done to research how the different task execution platforms affect the worker's performance and the task output quality. A reliable experimentation method will have to be found in order to draw useful conclusions.

1.3. Research question

To tackle the challenges we need to come up with a design and implementation which facilitates high quality task output and high worker satisfaction and engagement. Therefor the main research goal of this study is to implement a multi-platform urban object crowd-mapping application using street-level imagery. To achieve this goal, the following research question needs to be answered:

> **RQ:** How can we design a multi-platform crowd-mapping application for urban object mapping using street-level imagery?

To address the main research question we have four research sub-questions:

> **RQ1:** *What are the existing approaches for crowd-mapping urban objects?*
> To answer this question, a literature study will be conducted in which the history of crowd-mapping will be researched. This will result in a timeline of important crowd-mapping events and a taxonomy of techniques and strategies.

RQ2: *How can we build a platform which enables task-requesters to create and deploy urban object crowd-mapping tasks?*
This question will involve the design of a multi-platform crowd-mapping system. Based on the information gathered from the literature study, a proposal has to be created for the parts of the system such as task design and architecture. Furthermore, a workflow has to be developed which abstracts the concept of crowd-mapping.

RQ3: *How can object properties be sensed time- and cost-efficiently with high accuracy?*
To answer this question it will be studied how to evaluate the tasks with the previously made design decisions. This will result in a proposal for geo-location and height estimation techniques.

RQ4: *How do the implemented platforms affect the worker satisfaction and output quality for the different types of tasks?*
As the final part of this thesis an experiment will be conducted using a case study which will be discussed in the experimentation and evaluation chapters. This should give insight into the effect the different task execution platforms have on the worker satisfaction and engagement, as well as the output quality factors such as geo-location accuracy and verification quality.

1.4. Contribution

This thesis will yield several contributions. A multi-platform crowd-mapping application for urban object annotation using street-level imagery will be developed, using both techniques from previous research and using more novel approaches. This application will support a web-based, a mobile-based and a mobile virtual reality-based task execution. Furthermore a methodology will be provided on how to estimate or sense properties, such as geo-location and height, on these urban objects.

As a final contribution, a use-case will be analyzed. Experimentation will be done with the implemented task execution platforms, which will give insight into the accuracy, participation, execution time and user satisfaction across these platforms for different types of tasks. During this experimentation, the proposed geo-location estimation technique resulted in 83% of the estimated geo-locations being within 2.5 meters of the ground truth for the web platform with a mean distance of 1.85 meters. The height estimation showed up to 85% of the estimations being within 30 centimeters from the ground truth with a mean difference of 15 centimeters. We demonstrated the feasibility of executing mapping, data-enrichment and verification tasks on each of these platforms. Experimentation showed that the mobile platform had the lowest execution times with average execution times around 13 seconds for each of the task types. It also showed that the mobile VR tasks had a significantly higher perceived cognitive load for the participants when compared to the other platforms, whereas the satisfaction and user engagement were similar. However, the results also showed that the type of task does affect the user engagement and satisfaction.

2

Related work

To gain insight in the state-of-the-art in crowd-mapping, a literature study was done. First, the history of crowd-mapping is researched and summarized in a timeline. Additionally, an assessment is made of the existing crowd-mapping systems and their corresponding properties and strategies, which will be discussed in the taxonomy section of this chapter. Based on the this related work it is discussed where a research gap exists which this research aims to fill.

2.1. History of crowd-mapping

Crowd-mapping is a relatively new area of research. Traditionally, mapping is done by having workers physically explore the area to be mapped. This is a time-consuming and cost-inefficient approach [6], which called for alternative approaches as governments are no longer willing to pay these costs [12]. The history of the crowd-mapping concept is discussed in the following sections, resulting in the timeline of Table 2.1.

2.1.1. First occurrences

The first examples of crowd-mapping can be found in managing global crises and disasters, dating back as far as January 2008, where the Ushahidi[1] crowd-mapping platform was used to map post-election violence in Kenya [24]. Two years later, during the 2010 Haiti earthquake the Ushahidi platform was used to collect data about the location of events that happened as a result of the earthquake such as fires and collapsed buildings [34].

2.1.2. Definitions

The term crowd-mapping starts appearing in research in 2011 [25, 29, 36]. A definition is found in literature the following year by Caminha et al., where they define the concept as "combining the aggregation of a Geographic Information System and crowd-generated content" [8].

Another term that arose in 2011, is the concept of (mobile) crowd sensing: "individuals with sensing and computing devices collectively share data and extract information to measure and map phenomena of common interest" [10]. This concept capitalizes on the fact that most people own a mobile device with sensors like GPS, microphone, camera and compass. Using these sensors to collect mapping data could resolve some of the issues of traditional mapping. Collecting the sensor data from widely spread mobile devices no longer requires traditional workers to physically be in the area, as the crowd is already there. Crowd-mapping focuses on utilizing the crowd to collect geographic data and therefor crowd sensing is one of the alternatives to do so. One downside of the crowd sensing method is that it requires participants in the area which is to be mapped.

[1] https://www.ushahidi.com/

2008	Release of Ushahidi: First use of crowd-mapping during human-rights crisis [24].
2010	First use of crowd-mapping during natural disaster [34].
2011	First appearances of the term "Crowd-mapping" in literature [25, 29, 36].
2011	Crowd sensing defined in literature [10].
2012	Crowd-mapping defined in literature [8].
2012	Feasibility study for crowd-mapping using street level imagery [20].
2013	Elaborate research on crowd-mapping using street level imagery [14].
2014	Research on mapping by combining crowd-sourcing and machine learning [15].
2016	Start of pilot for crowd-mapping using street level imagery at scale [31].
2019	Elaborate research on crowd-mapping using street level imagery at scale [32].

Table 2.1: Timeline of crowd-mapping.

2.1.3. Using street-level imagery

Recent research has been trying to resolve this issue using street-level imagery [27]. Services like Google Street View[2], Mapillary[3] and OpenStreetCam[4] offer worldwide street-level images with high coverage. Using these images no longer requires the crowd workers to be in the area, as they can virtually look around and collect the required data. A first feasibility study was done by Kotaro et al. in 2012 where they looked at the possibility of using Google Street View to determine sidewalk accessibility issues [20]. They concluded that "untrained crowd workers can locate and identify sidewalk accessibility problems with relatively high accuracy (80% on average)" [20]. The next year they published the follow-up study [14] and more researchers published papers on combining street-level imagery and crowd-mapping [28, 33]. In 2014 Hara et al. experimented with combining the crowd-sourced data with machine learning techniques [15].

In September 2016 Saha et al. started a pilot with Project Sidewalk, another system to determine sidewalk accessibility issues using Google Street View which they introduced in a publication in 2017 [31]. This was the first pilot at scale for a crowd-mapping system using street-level imagery with 581 contributing users collecting 71,873 labels. In 2019 they published the results after a 18-month deployment of their system [32].

2.2. Automatic detection of urban objects

An alternative to crowd-mapping is using an automated approach to detect object in street-level imagery. Even though this is not the focus of this project, it is useful to look into some of these systems to see what the advantages and disadvantages are compared to crowd-mapping.

In recent years multiple systems have been developed with the aim of cataloging urban trees [21, 22, 37]. The classification algorithm by Wegner et. al. for example [37] uses multiple views to detect the geo-location and species of trees in an urban environment. Although their results are fairly accurate, they do indicate a number of false positives that are easily recognized by humans. False positives for example occurred when the tree was occluded by another object or when the algorithm classified a telephone pole as a tree due to the visual similarities. Furthermore, the system has to be trained using human-classified images. These are general issues with current computer vision techniques and due to these potential errors and the required training, crowd-mapping could be used to improve on these weaknesses.

2.3. Street-level imagery crowd-mapping taxonomy

Street-level imagery can be used to map a broad range of properties. Each of these properties require a specific strategy to accurately collect this data. In this section, related literature is examined to identify the mapped properties and corresponding strategies found by previous research.

2.3.1. Existing crowd-mapping systems

Currently existing crowd-mapping systems found in literature that utilize street-level imagery mostly focus on either mapping accessibility data (Table 2.2) or mapping cityscapes (Table 2.3). These systems with their corresponding strategies are outlined below.

2.3.2. Properties & strategies

The mapped properties are either objective or subjective. The objective properties can be categorized into the categories:

- Geometrical/morphological: information about for example the shape or geo-location of an object;
- Functional: the function of an object, i.e. "school" or "hospital";
- Material: describing what the object is made of;
- Landscape: the capacity of the area, i.e. the amount of parking spaces on a parking lot.

Other properties are subjective and are categorized as contextual, i.e. the safety or attractiveness of an area. Table 2.4 shows an overview of these property categories and the corresponding strategies to sense the properties.

[2] https://www.google.com/streetview/
[3] www.mapillary.com
[4] https://openstreetcam.org/

Property	Imagery service	Strategy
Sidewalk Accessibility Data [31, 32]	Google Street View	Let workers explore a predetermined route by following turn-by-turn directions, label accessibility issue by clicking the area of interest and categorizing into one of five types (i.e. "Surface problems"), assess severity (1-5) and leave optional notes.
Street-level Accessibility Data [14]	Google Street View	Three steps: draw outline of accessibility issue, categorize into one of five types (i.e. "Object in path") and assess severity (1-5). Also has verification task, ask if workers agree with assessment of the three steps.
Bus stop landmarks for blind riders [16]	Google Street View	Drop worker near bus stop in Street View, let worker label six types of landmarks (i.e. "bus stop sign") close to the bus stop by clicking the object of interest.
Street-level accessibility data for sidewalks, bus stops, and intersections [15]	Google Street View	Let worker explore area, when either a location with a curb or a location with a missing curb is found, the worker can draw an outline of this area in the image. Verification is also done by the crowd, by letting workers verify the outlined areas.

Table 2.2: Accessibility data crowd-mapping systems using street-level imagery

Property	Imagery service	Strategy
Geo-location of urban objects [27]	Google Street View	Let worker explore area, when an object of interest is found enter annotation mode and annotate by drawing a bounding box over said object.
Visual perceptions of quiet, beauty and happiness across London [28]	Google Street View	Show worker two images of different locations in London and ask which one of the two is the most quiet, beautiful or happy looking.
Perception of city locations by youth [30]	Images provided by researchers	Show an image to the workers and let them qualify the area by choosing from a seven-point range from strongly disagree to strongly agree for six descriptors: "dangerous", "dirty", "nice", "conserved", "passable" and "interesting" [30].
Perception of safety of city locations [33]	Google Street View and images provided by researchers	Show worker two images of different locations and ask the worker to pick the image which looks safer.

Table 2.3: Cityscape data crowd-mapping systems using street-level imagery

	Geometrical/morphological	Functional	Landscape	Contextual
Numerical/range assessment				Street-level Accessibility Data [14], Sidewalk Accessibility Data [31, 32], Perception of city locations by youth [30]
Bounding box drawing	Urban objects [27]			
Outline drawing	Street-level Accessibility Data [14]			
Clicking area of interest	Sidewalk Accessibility Data [32], Bus stop landmarks for blind riders [16]			
Text labeling	Sidewalk Accessibility Data [32]	Sidewalk Accessibility Data [32]	Sidewalk Accessibility Data [32]	Sidewalk Accessibility Data [32]
Image alternative selection				Visual perceptions of quiet, beauty and happiness across London [28], Perception of safety of city locations [33]

Table 2.4: Strategies for different crowd-mapping properties as implemented by previous research

2.4. Research gap

Based on the studied literature, it is recognized that a research gap exists. Firstly it became evident that crowd-mapping can not yet be replaced completely by computer vision techniques as they still produce false-positives which would easily be recognized by humans. Furthermore, these computer vision systems need to be trained by human annotated data. All of the existing crowd-mapping systems found in the literature study are developed on web-based platforms. No research has been done on executing crowd-mapping tasks on alternative platforms and how this would affect output quality and engagement. All systems also focus on a single type of object (trees, sidewalks, etc.) and do not make an abstraction of the concept of crowd-mapping, with the exception of the work by Qiu et. al. [27] which serves as a proof of concept, but also mentions that further research needs to be done on this subject. One of the aspects is the geo-location estimation based on the workers input, which remains a challenge for all of the studied systems. Finally, none of the systems estimate the height of the annotated objects.

2.5. Summary

A literature study was conducted which showed the origins can be found in managing crises and disasters in 2008 when the Ushahidi system was developed to map post-election violence in Kenya. In 2010 this platform was used to collect data about events surrounding the Haiti earthquake. In the following years new crowd-mapping systems would emerge and in 2011 the concept was defined as "combining the aggregation of a Geographic Information System and crowd-generated content" [8].

Soon thereafter, in 2012, experimenting began with combining crowd-mapping with street-level imagery, when Kotara et. al. conducted a feasibility study [20]. After they deemed this experiment successful, they released a follow-up study a year later [14] and more studies followed [15, 28, 33].

A taxonomy was made for crowd-mapping using street-level imagery in which the existing systems could be categorized as either an accessibility data crowd-mapping system or a cityscape data crowd-mapping system. The main focus of the first type is mapping accessibility issues in cities for people with disabilities, whereas the latter focuses on mapping objects and perceptions of cities and their surroundings. These systems all map properties, which are either subjective or objective, using a certain strategy. The objective properties could be categorized as geometrical/morphological, functional, material or landscape and the subjective properties were contextual. The strategies for mapping these properties found in literature were bounding box drawing, outline drawing, clicking area of interest, text labeling and image alternative selection.

In the studied literature a gap was recognized in executing crowd-mapping tasks on platforms other than those that are web-based. Furthermore none of the systems abstract the idea of crowd-mapping in a manner that the system can be used for general purpose goals. Finally, geo-location estimation remained a challenge throughout the systems and none of them estimated the height of the mapped objects.

3

Design

3.1. Task design

To ensure high output quality for the tasks executed by workers, a appropriate task design is required. The find-fix-verify pattern as proposed by Bernstein et al. separates the responsibility of each task executed by the crowd. This pattern splits tasks into a series of generation and review stages, with the aim of improving the task output quality: identification, generation, and verification stages [7]. In their research Bernstein et al. showed that this pattern could achieve high quality task output despite relatively high individual error rates [7]. This means that even though there might be workers that execute certain tasks with errors, this will be fixed in different stages of the find-fix-verify pattern. For each of these types of tasks, a translation is made for crowd mapping which is discussed in the following sections.

3.1.1. Find task

The find task is the first stage of the find-fix-verify pattern, which is the identification stage. For crowd-mapping this translates into the main objective being finding the object matching the description as specified by the task requester. The output of these tasks will be a list of geo-locations (latitude, longitude) of objects marked by the crowd worker. The quality of this output may vary, depending on the worker. For example errors could occur when a worker marks the wrong object or incorrectly marks an object. Therefor it has to be verified by other workers, which will be done in the fix and verify tasks.

3.1.2. Fix task

The second part of the find-fix-verify pattern is the generation stage. Based on the output of find tasks, fix tasks will be generated. The aim of the fix tasks is to check if the marked object is correct and, if so, to enrich the data generated at the find task. When the find task worker has made an error, this result can be discarded by the worker executing the fix task. The output of this task will be a bounding box enclosing the marked object and a list of user generated labels. The bounding box may be used to output an image of the object which, combined with the labels, for example could help with machine learning training purposes.

3.1.3. Verify task

The final type of task serves as a quality control mechanism for the data generated at the find and fix tasks: the verification stage. For this task, the crowd worker will indicate whether the marked object matches the description and whether the bounding box correctly encloses the object. Finally, the worker will check if the user generated labels are relevant for the given task. When a certain object went through the find and fix process and was finally verified at the verify task, the collected data is aggregated and presented to the task requester. Optionally, a verify task can be executed by multiple workers, as it is also possible for workers to make errors during the verify task. The output will then be based on the majority of the votes for each verification part.

3.2. Platforms

An important part of this research is looking into different ways of executing crowd-mapping tasks. To facilitate this, a number of task execution platforms will be implemented. The reasoning behind these platforms is discussed below.

3.2.1. Web

A web-based platform could be considered the traditional approach for executing crowd-mapping tasks. All of the systems discussed in Section 2.3.1 are web-based. Implementing a web-based execution platform will give insight into the differences between a traditional approach and the more novel approaches of the other platforms.

3.2.2. Mobile

The vast majority of people in developed countries own a smartphone and adoption continues to grow. In 2019 there were an estimated 3.2 billion smartphone owners [4]. Using this platform for crowd-mapping task execution should therefor not be overlooked, also considering the fact that it has been used previously for crowd-sourcing in other studies for tasks such as digitizing local-language documents [13], surveys [9] and image tagging [38]. The difference in interaction as compared to web could bring challenges and possibilities. A significant advantage of this platform is the fact that smartphones are portable, which means tasks could theoretically be executed wherever and whenever.

3.2.3. Mobile VR

In recent years, virtual reality has become increasingly popular as more and more consumer products have been released by companies such as Oculus (as seen in Figure 3.1b) and HTC. It offers a much more immersive experience for gaming, as it gives the player impression the he is part of the game world. This aspect would also be interesting for crowd mapping, as this gives the ability to roam around the world without actually physically being there. For a crowd mapping platform to be useful however, it needs to be scalable and be accessible by a large group of people, which is not the case for the dedicated virtual reality headsets as they are still relatively expensive. For 2020, the forecast is that the installed base of virtual reality headsets will grow to 37 million [5]. This number shrinks into insignificance compared to the 3.2 billion smartphone users. An alternative to dedicated virtual reality headsets is to use the screen and processing power of the mobile phone to create a virtual reality experience. Projecting a stereoscopic image on the phone screen and using a pair of lenses is much more cost effective and accessible. These lenses with phone holders are widely sold, by for example Google with their Cardboard (as seen in Figure 3.1a) which is, as the name suggests, made out of cardboard and therefor very cheap. Another advantage is that this is a portable solution, so tasks can be executed wherever you are, as long as you have your phone and the virtual reality goggles. Research has been done on using virtual reality for crowd-sourcing tasks, such as the study by Ma et al. [23] in which they used dedicated virtual reality headsets such as the HTC Vive and Oculus Rift, as well as mobile virtual reality devices such as the Google Cardboard and Samsung Gear VR. In this study they showed that executing crowd-sourcing tasks using virtual reality is feasible. Research aimed specifically at crowd-mapping is, to the best of our knowledge, still non-existent however.

3.3. Architecture & Techniques

As the crowd mapping platform should support multiple task execution platforms, the choice was made to implement an application programming interface (API). This allows access to the service using network requests and therefor can be used by a wide variety of platforms and programming languages. This API can for example be used to retrieve tasks and submit tasks results. Three task execution platforms were implemented initially, as discussed in the previous section, but having an API gives the possibility for implementation of other platforms in the future.

The API is used to communicate with the backend. The backend of the system is implemented using a loosely-coupled, dependency-inverted architecture. This architecture consists of four main components: the API, a core, the infrastructure and a shared kernel, which is visualized in Figure 3.2. This pattern decouples the data access logic from the business logic. Data access methods are defined in interfaces in the shared kernel, which are implemented in the infrastructure. The API layer then only references the interfaces from the shared kernel (dependency-inversion), leaving it "agnostic" as to where the data comes from. This sepa-

(a) Google Cardboard mobile VR device. Source: [2]

(b) Oculus Rift S dedicated VR device. Source: [3]

Figure 3.1: Mobile VR device and dedicated VR device.

ration of concerns is considered as good practice by the SOLID principle [1]. Furthermore, this pattern allows for implementation of dependency injection, which improves the test-ability of the system. The backend is implemented in C# .NET Core with the Entity Framework.

The API contains endpoints which can be called to perform actions in the backend. The requester dashboard is used to create new crowd-mapping tasks and analyze the output by these tasks. The task execution platforms use the same API but call the endpoints for starting tasks and submitting results. The Core component contains all the entities (models) related to creating and executing tasks and executing the corresponding results. Furthermore it contains logic for authentication and other task creation/submission related operations (for example generating images with bounding boxes), summarized as "Business logic". Finally, the Infrastructure component is responsible for the data access operations. This includes saving tasks/task results to the database as well as fetching this data from the database.

[1] https://itnext.io/solid-principles-explanation-and-examples-715b975dcad4

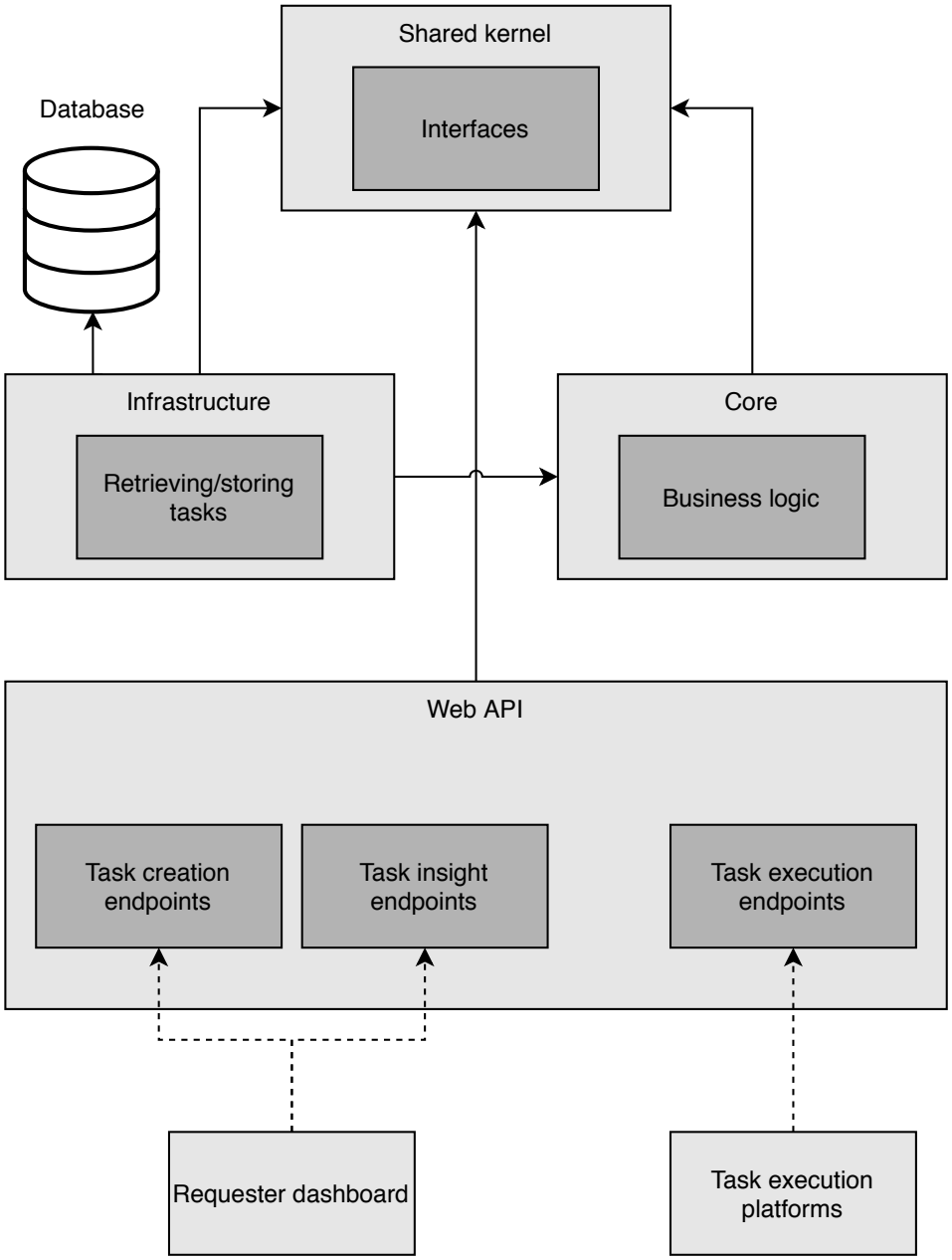


Figure 3.2: Crowd-mapping application system architecture.

3.3.1. Web and mobile platform

The web and mobile platform are implemented using the Ionic framework[2], which is a closs-platform app development platform. This means they both run on the same code base, but with slightly different presentations and interactions. The advantage of using such a framework is that it cuts down the development time, as the code only has to be written once. An important consideration during development however was that screen sizes may vary significantly between web app users and mobile app users. The application needs to scale correctly to every screen size, such that it is still convenient to use. Furthermore, the means of input differs, as the web app is controlled by mouse and keyboard and the mobile app is controlled using the touch screen of a mobile phone.

[2]https://ionicframework.com/

3.3.2. Mobile VR platform

The application is developed using the Unity software[3], which is game development platform with tools to develop for mobile VR. The panorama image is fetched from the street level imagery provider and wrapped on the inside of a sphere in 3D space. The user is then placed at the center of this sphere, with the ability to look around. This gives the impression you can look around in the space just as you would in real life. This induces the effect of place illusion [11], which is defined as " the strong illusion of being in a place in spite of the sure knowledge that you are not there" [35].

A consideration while implementing this platform is that the user will have no or limited ways of interacting with the phone, as it is not physically accessible while in the goggles. Some goggles have buttons which can be used to interact with the device, but as this is not the case for all of them, the app needs to be controlled without physical interaction with the device itself. This interaction is elaborated on in Section 4.3.2.

3.4. Workflow

Both the task requesters and the crowd workers follow a specific workflow. The workflow for both of these users is explained in the following sections.

3.4.1. Task requester

The workflow for a task requester can be seen in Figure 3.4. When a requester want workers to execute a crowd-mapping task, he first needs to create this task. To create a task, the requester will use the task creation dashboard, where he will be asked to specify the following details for the task:

- **Task goal**: Specify the goal of the task to be create by entering a title and a short description of what is expected from the crowd workers
- **Area bounds**: The area in which workers will look for the object as specified in the task goal
- **Budget**: For each verified task result, workers will receive a reward. The task requester will have to specify a total budget for the task

After providing these details, the task can be deployed for the workers to execute. For each of the task types (find/fix/verify) data is collected, which can be viewed in the dashboard. When all tasks have been completed by the workers, this data is aggregated and presented to the task requester for it to be analyzed.

[3] https://unity.com/

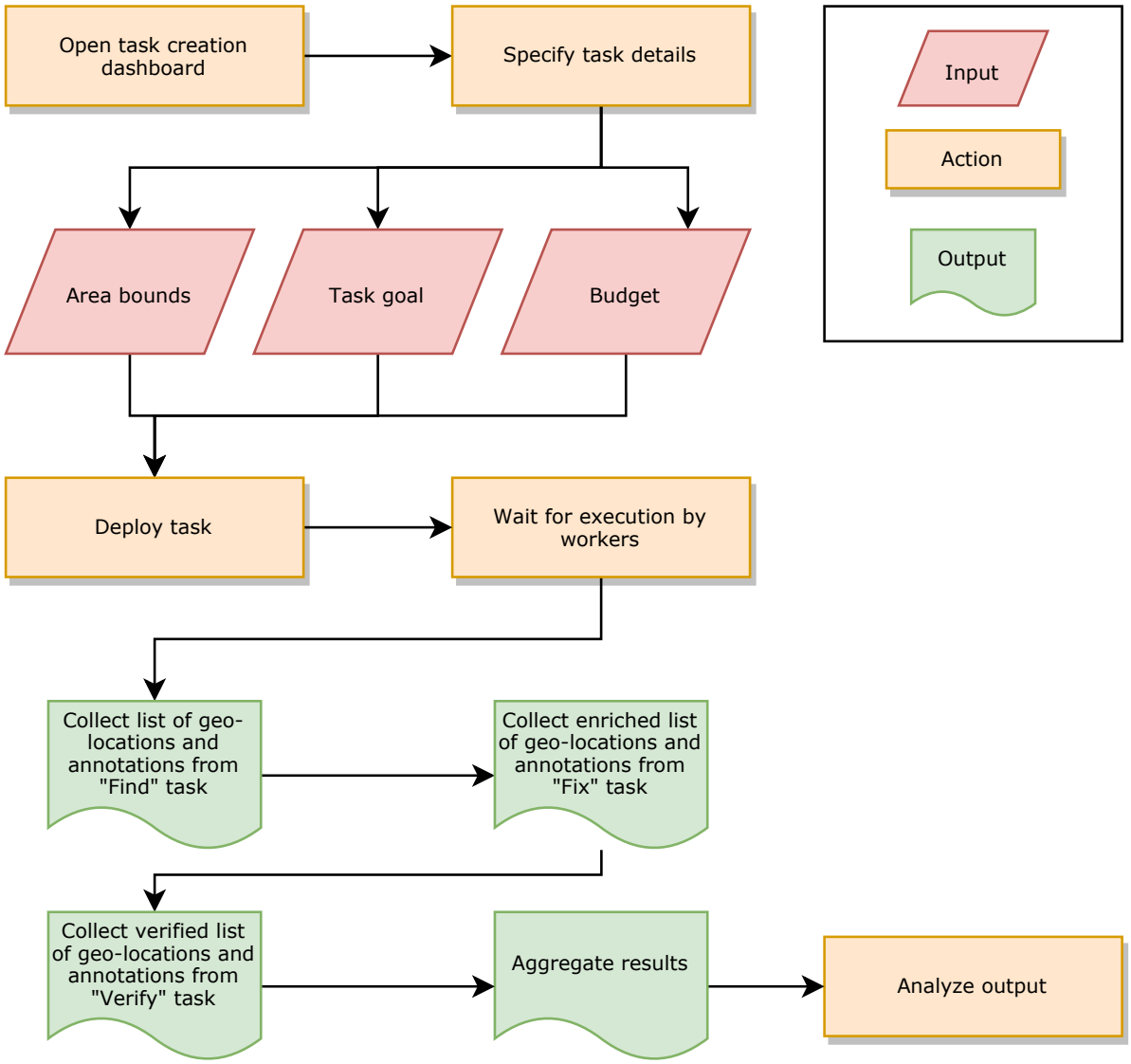


Figure 3.3: The workflow for a task requester.

3.4.2. Crowd worker

When a requester deploys a task as explained in the previous section, it is ready to be executed by the crowd worker. When a worker request a task from the system, it will receive a find, fix or verify sub-task. Initially, only find tasks are created for the deployed task. For this task type the user is assigned to a certain part of the area bounds as specified by the task requester, where he is free to roam around. When an object is marked the geo-location of the object is calculated, as will be explained in Section 3.5. Multiple objects can be marked in a single find task, which the user is then able to submit to the system.

For each found object in the find task, a fix task is created. Whenever the user requests a task and receives a fix task, he is presented with the object as found at the find task by another user. The user will be free to look around in this specific street level panorama, but is not able to roam around. When the find task worker has made an error and has not marked a correct object, the task can be discarded by the fix task worker. If the object marking is correct however, a bounding box enclosing the object will then need to be drawn and additionally, the user is asked to provide at least one text label relevant to the object. These labels and the position of the bounding box is then submitted to the system using the API. Additionally, based on the bounding box drawn by the worker, the height of the object is calculated.

Finally, for each fix result that is submitted (combination of a bounding box and one or more text labels), a verify task is generated. Here the user is asked to confirm that the marked object indeed matches the task description, as well as verifying whether the bounding box correctly encloses said object. Furthermore, for

each label it has to be specified whether or not they are relevant for the marked object. After each task, the worker gets a reward and gets the option to start another task.

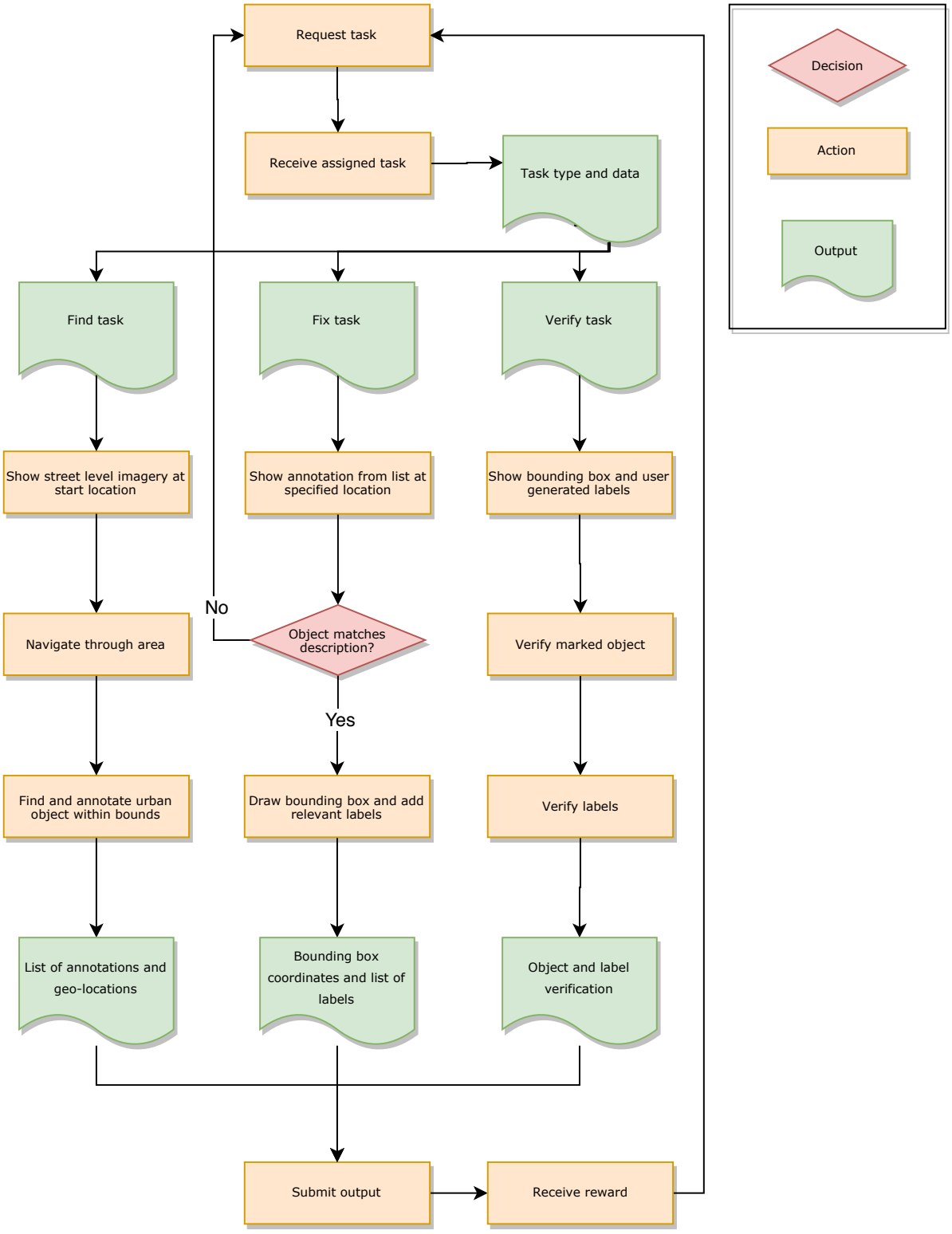


Figure 3.4: The workflow for a crowd worker.

3.5. Task evaluation

Two of the important output variables of the crowd-mapping tasks are object geo-location and object height. These variables have to be calculated based on the workers input. The methodology used for this is discussed in the following sections.

3.5.1. Object geo-location estimation

When executing the find task, users will be asked to mark a specific object in the task area. One important piece of data that needs to be extracted from this marking is the geo-location of said object, defined by the longitude and latitude. There are a number of known variables exposed by Google Street View whenever a user marks an object:

- Latitude and longitude value of the current street-level panorama image, indicating the location the panorama photo was taken at;
- The heading of the center of the viewport in which the panorama is shown in degrees relative from true north;
- The pitch of the center of the viewport in which the panorama is shown from -90 degrees (straight down) to 90 degrees (straight up).

Google collects depth data when creating panorama's, which can be accessed through undocumented API's. Project Sidewalk [32] uses this data to determine what the distance is between the camera and a certain pixel of the panorama. As this data is not available through Google Street View's public API, this approach is not suitable for us. Furthermore, this would make switching to a different street level imagery provider in the future more difficult.

Therefor it has to be estimated in some other way as this cannot be done accurately from a single perspective without depth information. The methodology used by Qiu et al. consists of casting a ray from the camera and based on the camera pitch and heading. It is then calculated where this ray would collide with the earth [27]. This approach however involves a simplification regarding the estimation of the height of the camera. As this height might vary and this information is not provided by Google, accuracy might suffer as a result. Therefor, we opted for an alternative approach.

When the same object gets marked from two positions however, an estimation of the geo-location can be made by figuratively drawing a line through the object from both perspectives and calculate where the lines intersect. To achieve this, it is not sufficient to draw a straight line from the perspective of the camera, as this would not take the curve of the earth into account. Therefore, the latitude and longitude of both perspectives first have to be projected in such a way that a straight line from both points corresponds to a curved line on earth. An example of such a projection is shown in Figure 3.5.

In this top-down view, point A and point B are used to calculate point C. After calculating the intersection and obtaining point C, the projection for this point is reversed which results in a latitude and longitude of the object. For both point A and B the latitude lat, longitude lng and heading h are known. The projection is done by assuming the earth is a perfect sphere as follows:

- First translate the heading from the WGS (World Geodetic System) coordinate system to the Cartesian coordinate system:

$$h_0 = \frac{(90 - h_A) \mod 360}{180} \times \pi \,, \tag{3.1}$$

$$h_1 = \frac{(90 - h_B) \mod 360}{180} \times \pi \,. \tag{3.2}$$

- Set (lat_A, lng_A) as the origin (x_1, y_1) of the Cartesian coordinates :

$$x_1 = 0 \,, \tag{3.3}$$

$$y_1 = 0 \,. \tag{3.4}$$

- Then set x_2 and y_2 to the equation representing the line from (x_1, y_1) with heading h_0

$$x_2 = cos(h_0) \,, \tag{3.5}$$

$$y_2 = sin(h_0) \,. \tag{3.6}$$

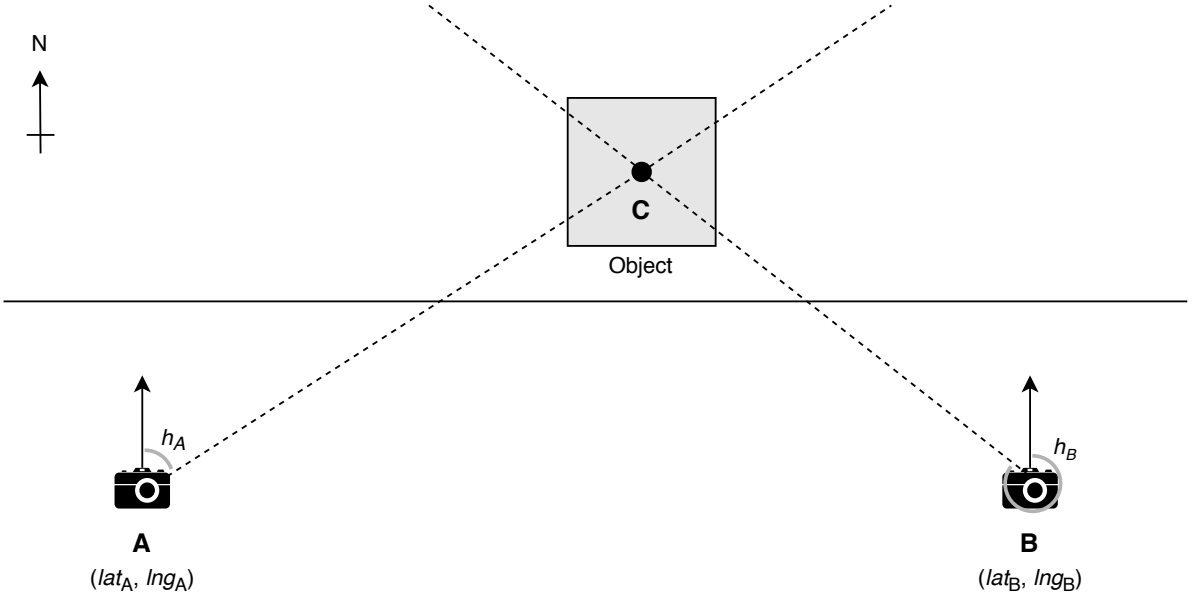


Figure 3.5: Location estimation by intersection of lines in a given heading

- Translate (lat_B, lng_B) from WGS coordinate system to Cartesian coordinate system using the fact that each degree of latitude will be approximately 111300 meters:

$$x_3 = (lng_B - lng_A) \times 111300 \times cos(\frac{lat_A}{180} \times \pi)\,, \tag{3.7}$$

$$y_3 = (lat_B - lat_A) \times 111300\,. \tag{3.8}$$

- Then set x_4 and y_4 to the equation representing the line from (x_3, y_3) with heading h_1:

$$x_4 = x_3 + cos(h_1)\,, \tag{3.9}$$

$$y_4 = y_3 + sin(h_1)\,. \tag{3.10}$$

- This gives the projection of point A and point B, with a line drawn in their corresponding projected headings. Then calculate the intersection (x, y), representing the projection of point C:

$$x = \frac{(x_1 \times y_2 - y_1 \times x_2) \times (x_3 - x_4) - (x_1 - x_2) \times (x_3 \times y_4 - y_3 \times x_4)}{(x_1 - x_2) \times (y_3 - y_4) - (y_1 - y_2) \times (x_3 - x_4)}\,, \tag{3.11}$$

$$y = \frac{(x_1 \times y_2 - y_1 \times x_2) \times (y_3 - y_4) - (y_1 - y_2) \times (x_3 \times y_4 - y_3 \times x_4)}{(x_1 - x_2) \times (y_3 - y_4) - (y_1 - y_2) \times (x3 - x_4)}\,. \tag{3.12}$$

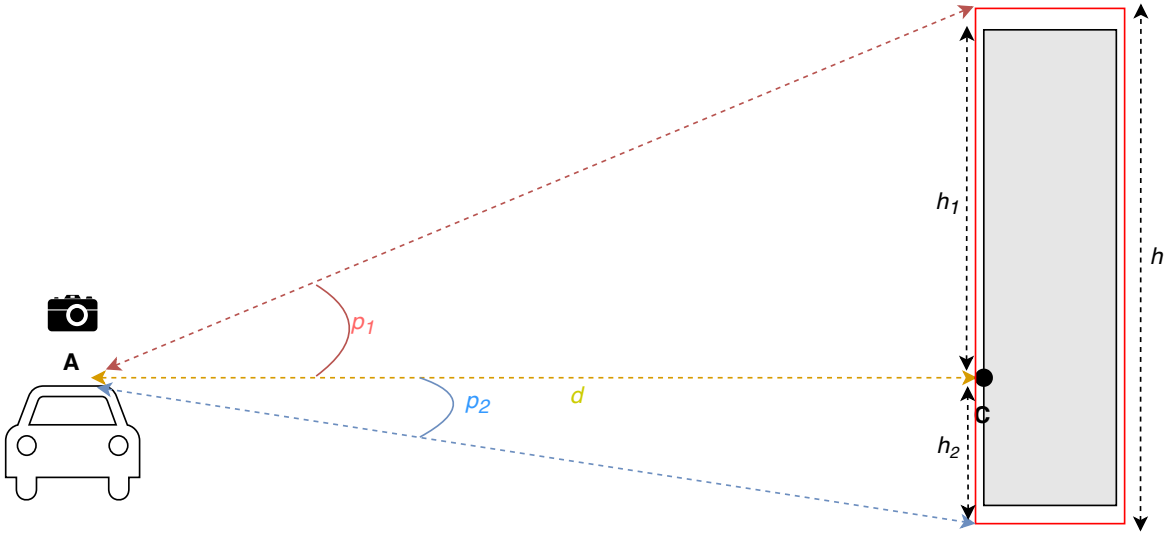
- With the projected coordinates of point C, the final step is to reverse the projection, which obtains the desired longitude and latitude of point C:

$$lat_C = \frac{lat_A + y}{111300}\,, \tag{3.13}$$

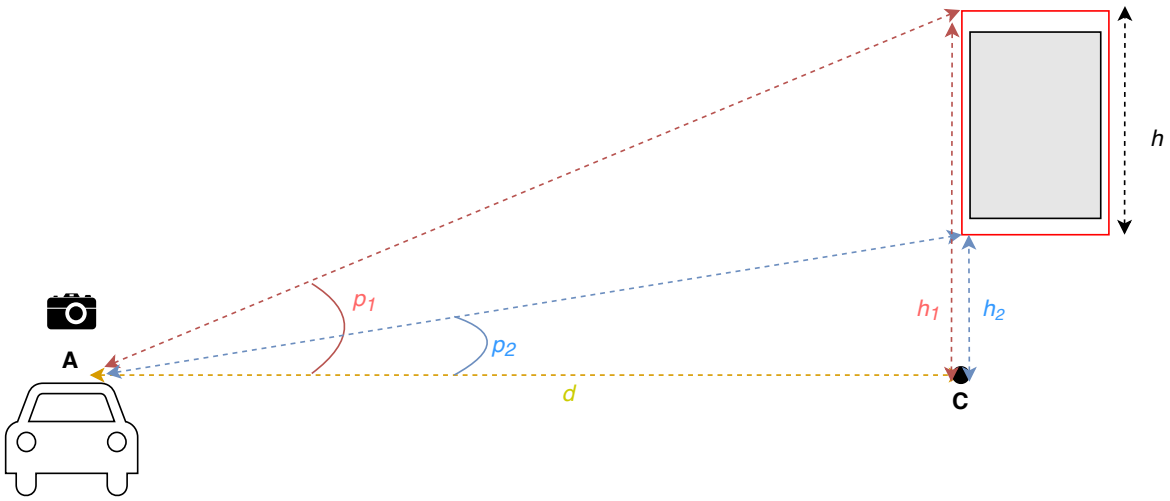
$$lng_C = \frac{lng_A + \frac{x}{111300}}{cos(\frac{lat_A}{180.0} \times \pi)}\,. \tag{3.14}$$

3.5.2. Object height estimation

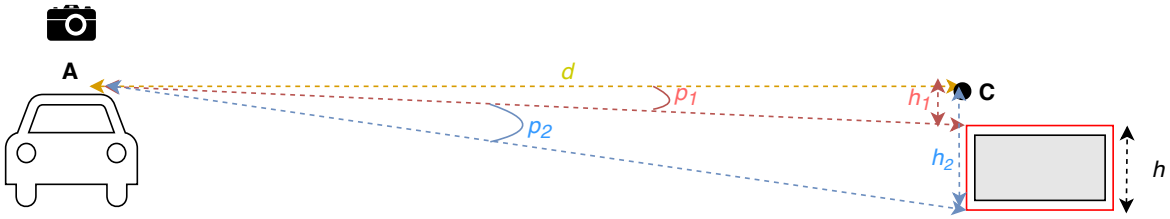
Another useful piece of information that can be extracted from the task results is an estimation of the height of the marked object. This is done by combining the estimated location of the object as found in the find task and the bounding box which is drawn at the fix task. There are three possible scenarios depending on the placement of the object relative to the camera. This has been illustrated in Figure 3.6, which is are side-views of perspective *A* in Figure 3.5.



(a) Scenario where top of object is above camera and bottom is below camera



(b) Scenario where top and bottom of object are above camera



(c) Scenario where top and bottom of object are below camera

Figure 3.6: Different scenarios height estimation by two geo-locations and known pitches.

The crowd sourced bounding box is depicted in red. The known variables are the latitude lat and longitude lng of both point A and C, as well as the pitch p_1 of the top of the bounding box and the pitch p_2 of the bottom of the bounding box. As the marked object might be elevated (scenario b) or lower than the camera height (scenario c), these pitch values can be either positive or negative.

To calculate the height h of the object, the distance d between point A and point C needs to be calculated using the Haversine formula:

$$\begin{aligned} \Delta lat &= |lat_A - lat_C| \,, \\ \Delta lng &= |lng_A - lng_C| \,, \\ a &= sin^2(\Delta lat/2) + cos(lat_A) \times cos(lat_C) \times sin^2(\Delta lng/2) \,, \\ c &= 2 \times atan2(\sqrt{a}, \sqrt{1-a}) \,, \\ d &= R \times c \,. \end{aligned} \tag{3.15}$$

With R the earth's radius of approximately 6371km. With the distance d between A and C known, calculating the length of h_1 and h_2 is a matter of calculating the side of a right-angled triangle:

$$\begin{aligned} h_1 &= tan(p_1) \times d \\ h_2 &= tan(p_2) \times d \end{aligned} \tag{3.16}$$

However, given that either one or both of the pitch values may be negative given by the scenario of Figure 3.6, either h_1 or h_2 or both might be negative. The height of the object h is then given by the difference between h_1 and h_2:

$$h = h_1 - h_2 \tag{3.17}$$

3.6. Summary

In this chapter the task design was discussed where the choice was made for the find-fix-verify pattern which splits tasks into a series of generation and review stages [7]. Furthermore, three task execution will be implemented: Web, Mobile and Mobile VR. These execution platforms will be communicating with the backend using an API. One of the outputs of these tasks is the estimated geo-location and we propose a method which involves the annotation of an object from two angles and calculating the intersection of these annotations. Finally, a method was proposed for estimating the height of an object, by using this estimated geo-location and the bounding box drawn by the workers.

4

Implementation

This chapter discusses the implementation of the system, based on the design decisions discussed in the previous chapter. All steps of the task creation, task execution and task analysis process will be explained. The task creation and task analysis sections discuss the implementation of the task requester dashboard, followed by the task execution sections which discuss the execution of each of the task types on the web, mobile and mobile VR platforms.

4.1. Task creation

Task requesters can create tasks by using the crowd map dashboard. They can login to this web application using a e-mail address and password, after which they are directed to the home screen depicted in Figure 4.1. This shows an overview of the currently deployed tasks, as well as the tasks that are no longer active under the "archived" tab.

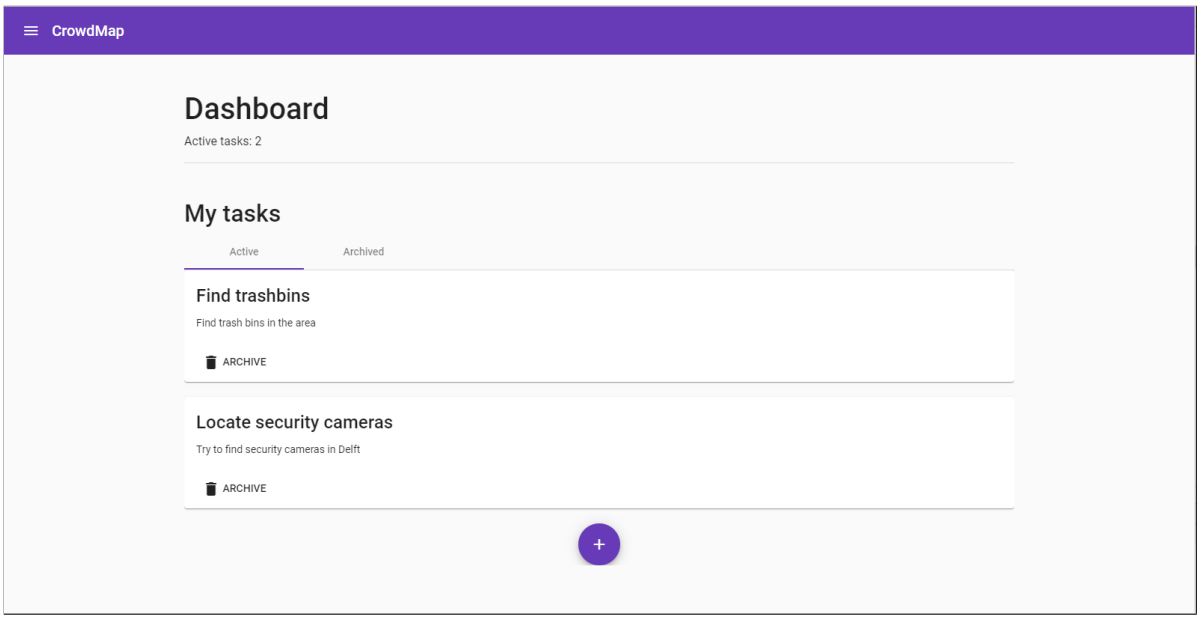


Figure 4.1: Task creation dashboard home screen.

From this screen, the user can either click on one of the tasks to see the results, or create a new task by clicking the "+" button. Doing this brings up the task creation form as seen in Figure 4.2. In this form all the details for the task will need to be specified. The area in which the task will take place can be specified by resizing a square over a map.

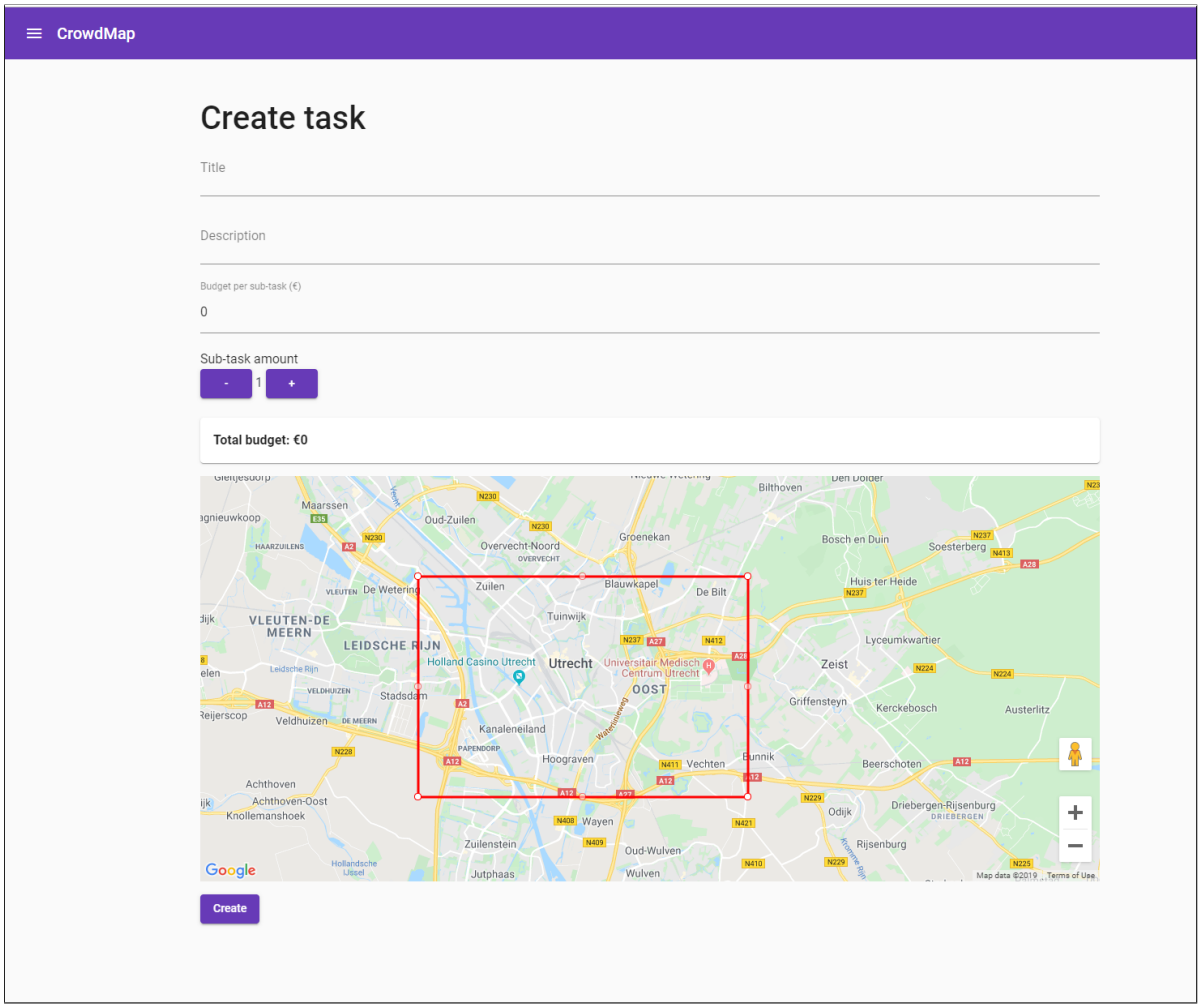


Figure 4.2: Task creation form.

4.2. Task analysis

When tasks are executed by workers, they submit results which are relevant for the task creator. These results can be analyzed in the crowdmap dashboard, as seen in Figure 4.3. They appear as markers on a map and are categorized in the three different phases of the task execution flow:

- Unverified results without bounding box - These are the results from a find task which have not been enriched by a fix task yet and also have not been verified by a verify task.

- Unverified results with bounding box and labels - The objects which have been enriched by adding a bounding box and labels are shown on the map with an orange marker. These results have not been checked by a verify task yet however.

- Completely verified results - Whenever a marked object has gone through the entire find-fix-verify flow, it is shown as a green marker on the map. This means it is an object which is considered to be accurate for the given task description.

When clicking a marker, a window appears with an image of the object and further details as shown in Figure 4.4. These markers can be toggled so the task creator can show only the desired results. Similarly the map shows a heat map of the found objects which can be turned on or off. This heat map gives insight into the density of objects in particular areas.

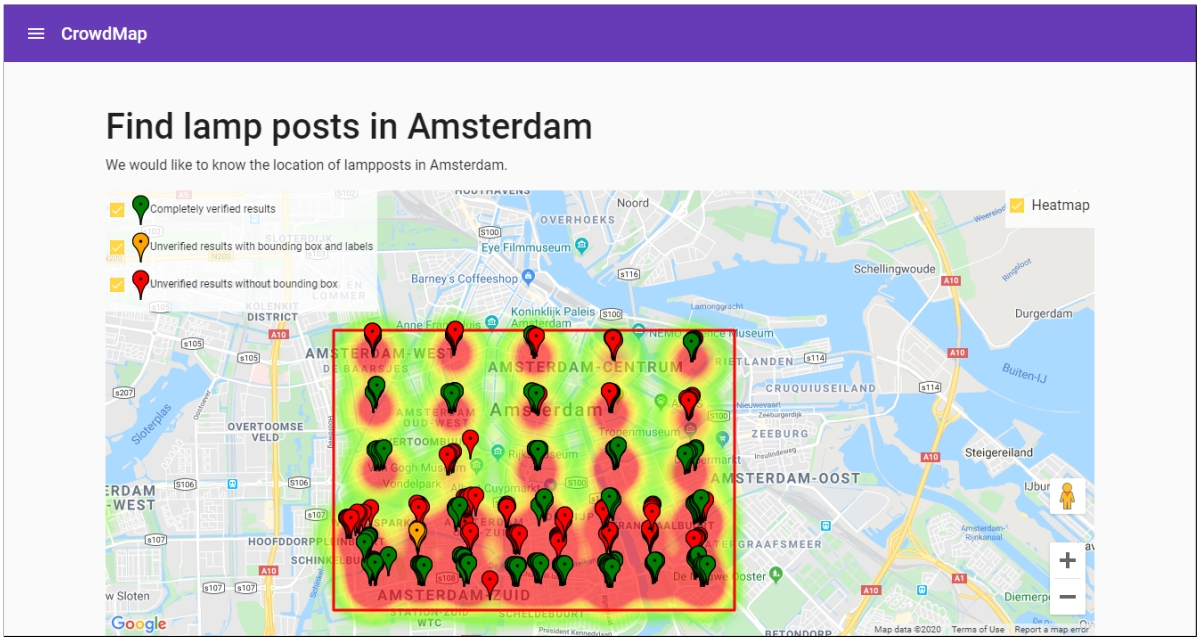


Figure 4.3: Task analysis screen.

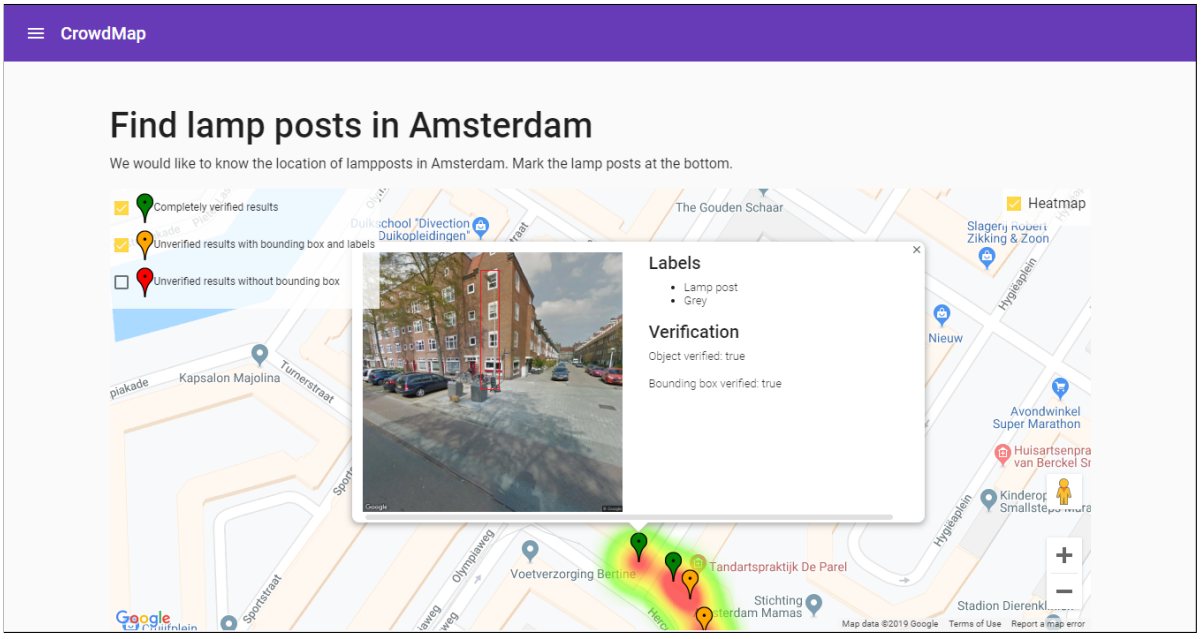


Figure 4.4: Task result details.

4.3. Task execution

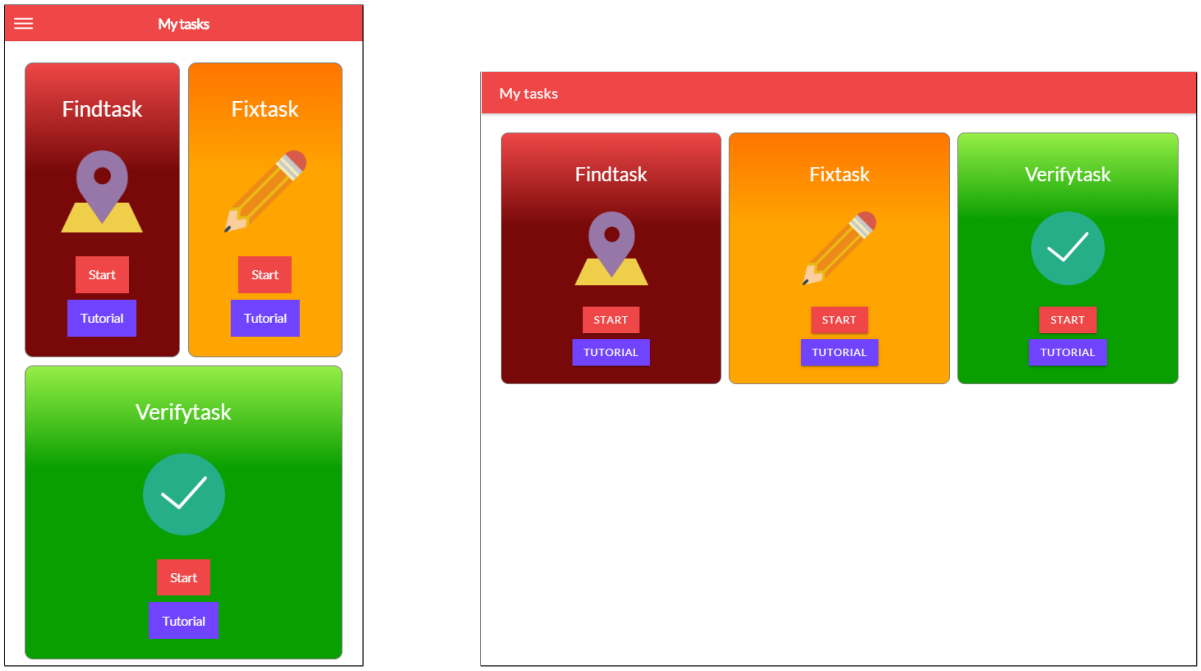
The crowd-mapping tasks can be executed on web, mobile and mobile VR. Each of the implementations are discussed in the following sections.

4.3.1. Web and mobile platform

As the web and the mobile platform are very similar, but mainly differ in terms of interaction and screen size, they are both showcased in the same section.

Interaction

The web and mobile platform differ in means of interaction, as the web platform is controlled by mouse and keyboard whereas the mobile platform is controlled by touch control. This was an important consideration when designing the task execution pages. For example it is harder to click accurately with a touchscreen as compared to clicking with a mouse cursor. Furthermore, the screen sizes differ so the this required a responsive user interface, which makes it scale correctly for every screen size. This is demonstrated in Figure 4.5 for the main menu of the application. In the following sections the implementations for the different task types are discussed.



(a) Mobile platform main menu

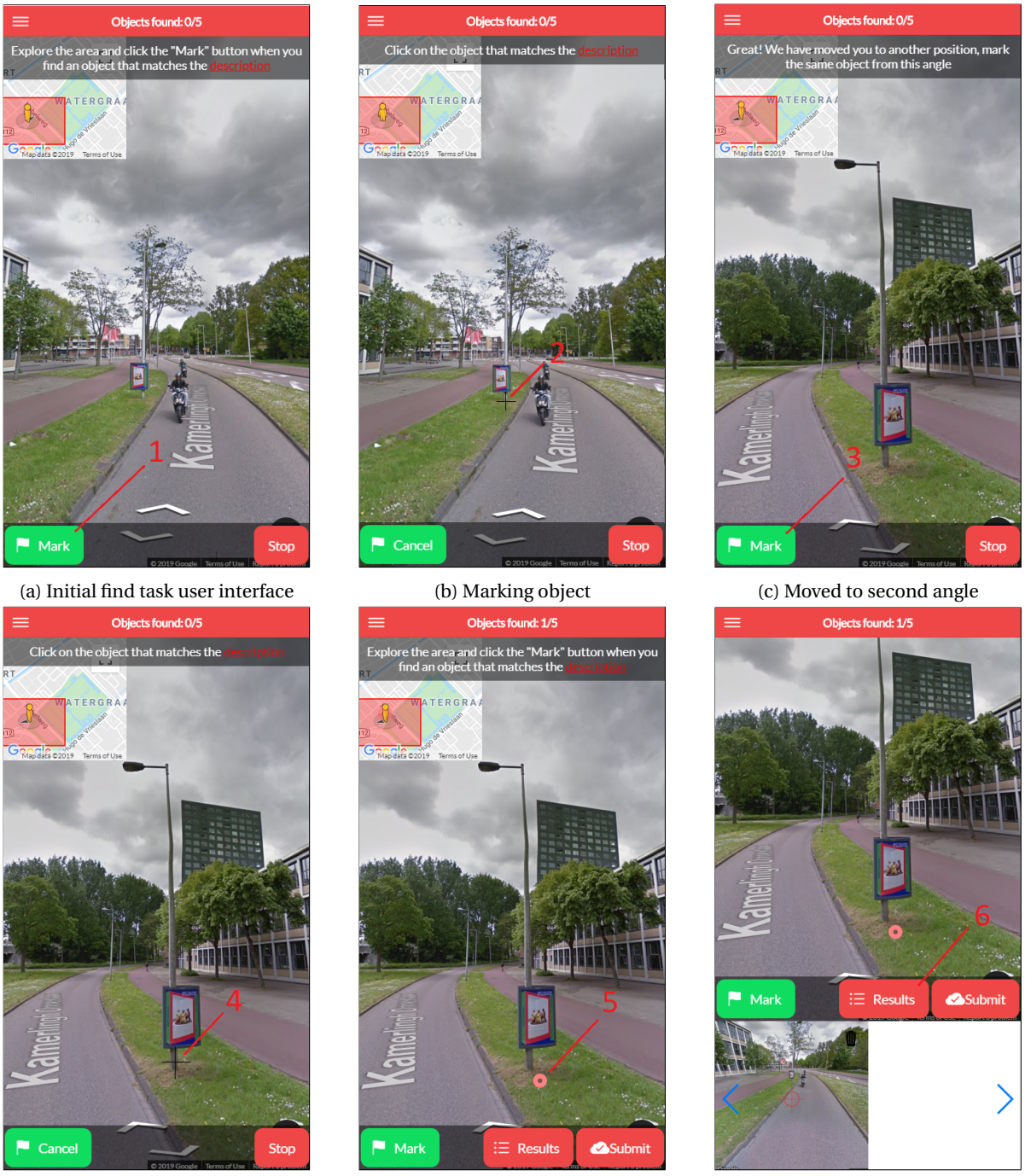
(b) Web platform main menu

Figure 4.5: Main menu user interface.

Find task

When starting a find task, the worker is placed in a location within the task area and is free to look and roam around the designated area. Looking around is done by clicking and dragging on web, or touching and dragging on mobile. Similarly, roaming around is done by clicking in the direction to move. Whenever the worker finds an object matching the task description, the marking flow can be started as illustrated in Figure 4.6:

1. First click the mark button to enter "marking mode";
2. Click on the object to mark with the cursor or by touch;
3. The worker is now automatically moved in the direction of the object to mark the object from the second angle as explained in Section 3.5.1;
4. Click the object again, as in step 2;
5. The geo-location of the object is calculated and a marker is placed on these coordinates;
6. The result can be viewed by clicking the results button and removed when required.



(a) Initial find task user interface

(b) Marking object

(c) Moved to second angle

(d) Mark object from second angle

(e) Object added to results

(f) Results overview

Figure 4.6: Find task marking flow.

Fix task

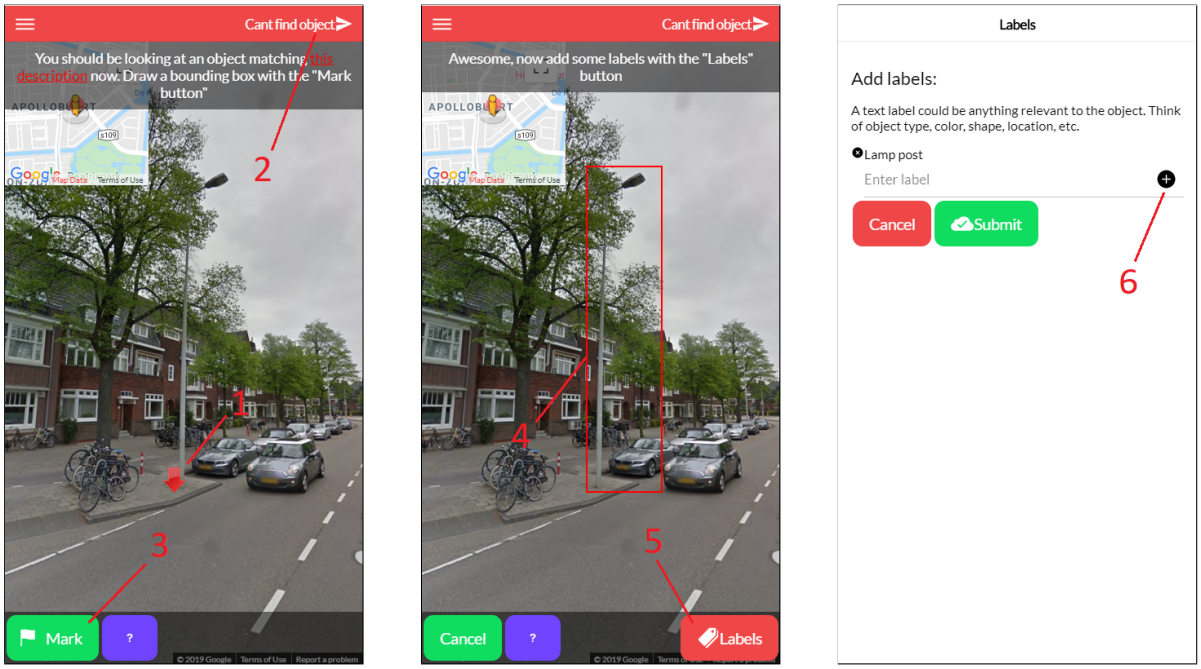
Workers who start a fix task are placed at a location where a previous worker marked an object. Here it is possible to look around, but it is not possible to roam around. The worker is then asked to enrich the result of the find task, as illustrated in Figure 4.7:

1. An arrow is placed at the object which was marked by the previous worker in the find task;
2. In the scenario where a worker incorrectly marked an object, the fix task executor can click the "Can't find object" button, which will set the marked object as invalid and close the current task;
3. If the worker does see the object, click the mark button to enter "drawing mode";

4. In this mode a bounding box can be drawn over the object by click and dragging on web and touching and dragging on mobile;

5. After drawing a bounding box, the "Labels" button appears which can be clicked to start adding labels;

6. Labels can be typed by the worker and added with the "+" button and removed by clicking the cross in front of a label.



(a) Initial fix task user interface (b) Drawing bounding box (c) Adding labels

Figure 4.7: Fix task enrichment flow.

Verify task

For the verify task, the worker is asked to check the object, bounding box and labels found in the find and fix tasks. This verification flow is shown in Figure 4.8:

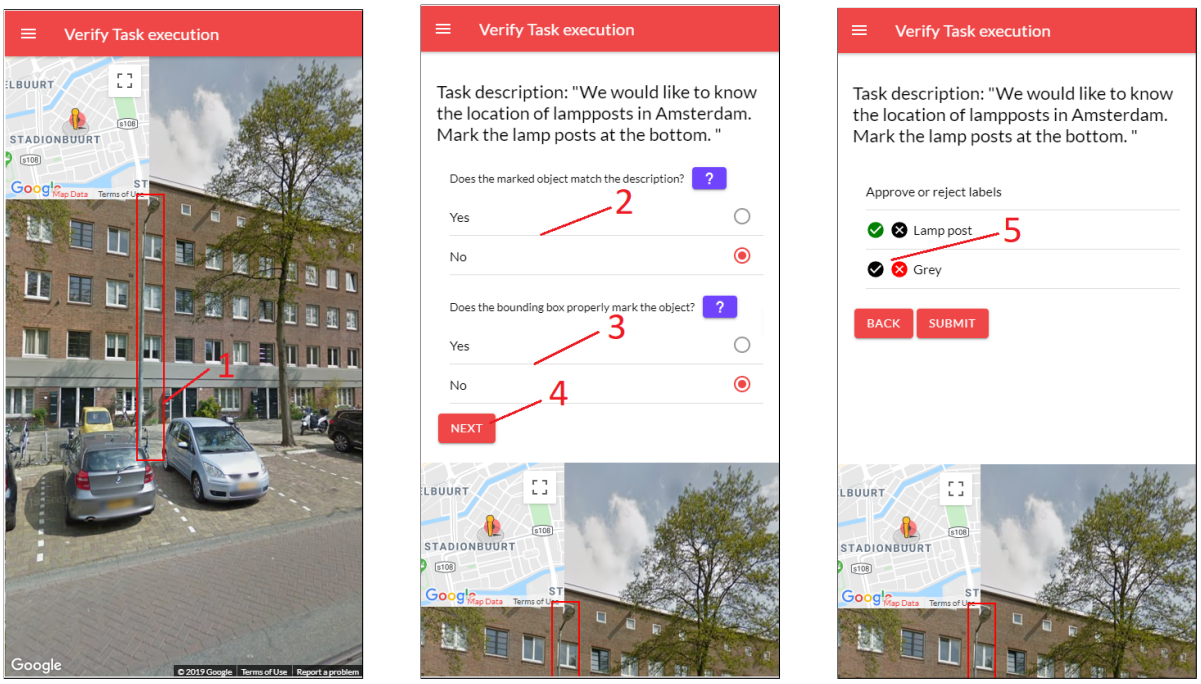
1. The object with the bounding box from the fix task is shown as an image;

2. The worker needs to verify whether the marked object does match the object described in the task description;

3. If the marked object is indeed what the task creator requested, the worker also needs to verify the bounding box;

4. After checking the object and it's bounding box the worker can continue to the label verification step;

5. When a label is considered relevant, the worker has to click the check mark in front of the label, if not the cross button needs to be clicked.



(a) Annotation by previous worker (b) Verification questions (c) Label verification

Figure 4.8: Verify task verification flow.

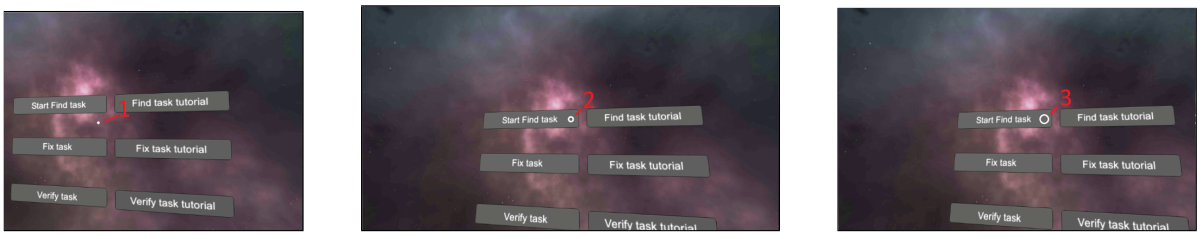
4.3.2. Mobile VR platform

The more novel execution platform is the mobile VR application. This implementation and the considerations are discussed in the following sections.

Interaction

As the mobile device is placed in the VR headset, interactions with the touchscreen are not possible. Therefor, it should be possible to interact with the application completely handsfree. Google has composed design guidelines for the Cardboard, where they advice the use of a reticle and a gaze-based UI [1]. This guideline was followed and the implementation is seen in Figure 4.9:

1. The reticle starts at its default size when no interaction takes place
2. When the reticle is in front of an item which can be interacted with, such as a button, it increases in size
3. Over time it keeps increasing in size and eventually reaches its maximum, after which the interaction is started. In the case of a button this would be a button click



(a) Initial reticle (b) Activated reticle (c) Component selected

Figure 4.9: Gaze-based interaction user interface.

This type of interaction is used throughout the application. Interactive items such as menu buttons are always placed in the virtual environment as opposed to being fixed to the screen, this is also considered best practice by aforementioned design guidelines. To expand the means of interaction, Bluetooth controller

support was also implemented, as well as voice command control. This gives the user the freedom of choice for interacting with the application.

Find task
A find task is started by gazing at the "Start Find task" button as seen in Figure 4.9. The flow the worker has to follow is illustrated in Figure 4.10:

1. The task description is placed "at the feet" of the worker in the VR space;
2. The worker can roam around until he finds an object matching the description. Movement is done by gazing at the green arrows, which transfers the worker to the next panorama;
3. When an object is found, the worker aims the reticle at it and marks it by either using the "mark" voice command, or clicking the mark button on the Bluetooth controller;
4. They are then automatically moved to an adjacent panorama to mark the second angle of the object in the same manner;
5. The object is then added to the results and when the worker has found enough objects these results can be submitted by using the "submit" voice command or clicking the submit button on the Bluetooth controller.

(a) Initial find task screen

(b) Marking object

(c) Marking object from second angle

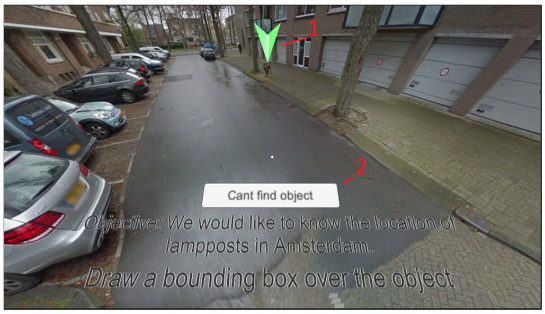
(d) Adding object to results

Figure 4.10: VR find task marking flow.

Fix task
When starting a fix task, the worker is placed in a panorama where another worker has previously marked an object. The worker then follows the flow illustrated in Figure 4.11:

1. The object that has been marked by the previous worker is indicated by a rotating green arrow;
2. It may happen that the previous worker has made an error and has not marked an object matching the description. In this case the worker can interact with the "Cant find object" button, which ends the task;
3. When the worker did find the object, it can enter "drawing" mode by using the "start" voice command or by clicking the start drawing button on the Bluetooth controller;

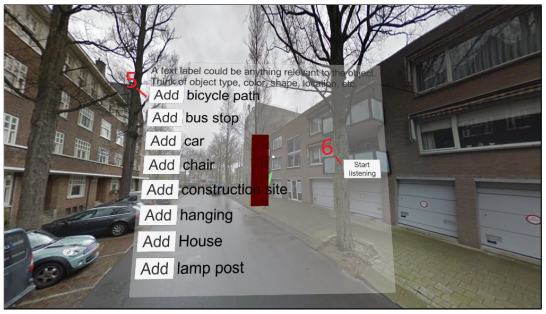
4. The worker can then draw a bounding box by using his head movement to change the dimensions and confirming by using the "stop" voice command or by clicking the stop drawing button on the Bluetooth controller;
5. The next step is to add labels, using the labels menu which is activated by using the "labels" voice command or by clicking the labels button on the Bluetooth controller. Here, eight of the most used labels by previous workers are shown, which can be added by interacting with the "Add" button;
6. When the worker wants to add a label which is not in the list of previous labels, he can add those using voice recognition. This is done by interacting with the "Start listening" button and speaking out the labels, with short pauses in between. Finally, when the bounding box is drawn and the labels are added, the results can be submitted by using the "submit" voice command or clicking the submit button on the Bluetooth controller.



(a) Initial fix task screen

(b) Start drawing bounding box

(c) End drawing bounding box



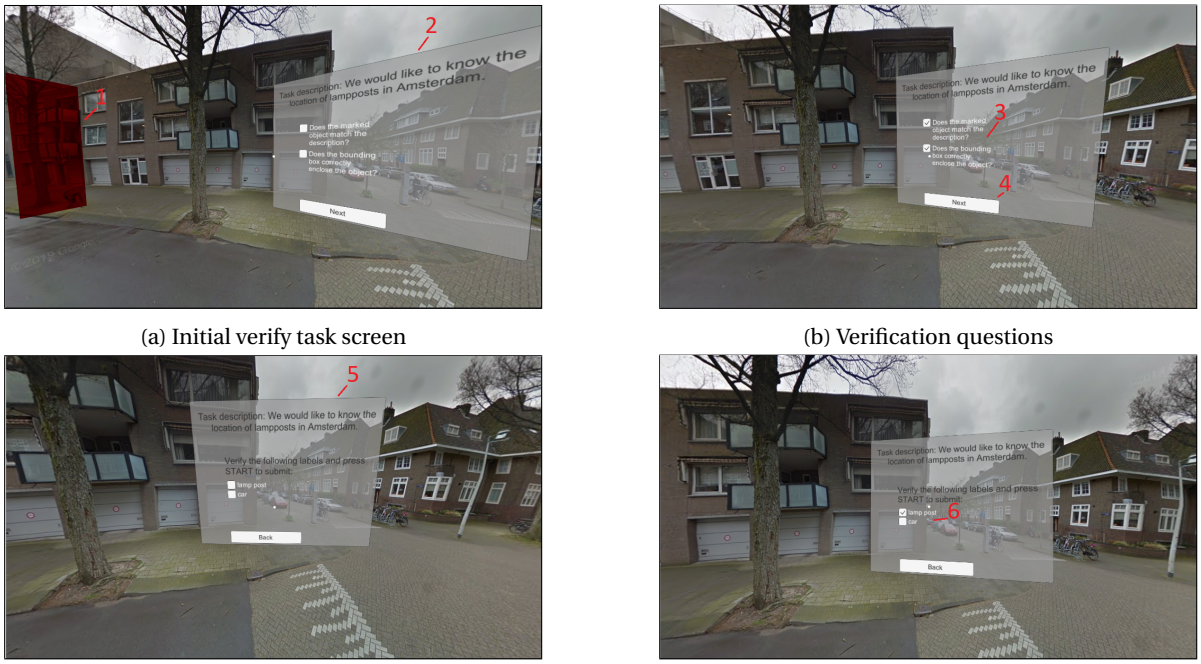
(d) Adding labels

Figure 4.11: VR fix task marking flow.

Verify task

The final task is the flow is the verify task, which can also be executed on the mobile VR platform. For this task, the worker is placed in a panorama where other workers have previously marked and enriched an object. The interaction is shown in Figure 4.12:

1. The bounding box which was drawn by a previous worker at a fix task is shown over the object;
2. Right next to the bounding box, the verify menu panel is shown;
3. The worker has to answer questions about the correctness of the object and the bounding box, by interacting with the checkboxes;
4. When these questions have been answered, the worker can continue to the next part of the verification flow by interacting with the "Next" button;
5. This brings up the label verification panel, where it shows the labels previously added by another worker;
6. These labels then need to be verified by interacting with the checkbox when the label is considered correct. Finally the results can be submitted by using the "submit" voice command or clicking the submit button on the Bluetooth controller.



(a) Initial verify task screen

(b) Verification questions

(c) Label verification

(d) Submitting result

Figure 4.12: VR verify task marking flow.

4.3.3. Tutorials

To familiarize workers with the controls, a guided tutorial was implemented for each task type on each platform. Here they have to complete a predefined routine which covers each aspect of the task execution flow, similar to Project Sidewalk [32]. This approach is commonly known as *onboarding*. As with Project Sidewalk, workers first have to complete the tutorial before they are allowed to start an actual task. This decreases the chance of workers making error caused by being unfamiliar with the task execution flow and interactions.

4.4. Summary

In this chapter it was discussed how the system was implemented using the design of Chapter 3. The task requester dashboard can be used to create a task by filling out the details in a form and selecting an area on the map in which the tasks will need to be executed. Furthermore they can analyze the results produced by workers on an interactive map.

The web and mobile implementations are similar but differ in interaction, as the web platform uses mouse input and the mobile platform uses touch input. The implementation for all of the task types on these platform was discussed in Section 4.3.1. The VR implementation required a user interface "in world space" and alternative means of interaction with gaze based controls, voice recognition and a Bluetooth controller. For all tasks and platforms, tutorial were implemented to train workers before they start executing actual tasks.

5

Experiments

5.1. Experiment plan

To evaluate the performance of the different platforms, a qualitative experiment will be conducted. The plan for this experiment and the variables involved will be discussed in the following sections.

5.1.1. Experimental procedure

Each of the participants will be assigned to one of the three task types (find, fix or verify). Furthermore each participant will execute tasks on each of the three platforms (mobile, web and mobile VR). To prevent bias as they learn the process, the order of the platforms used will be different for each participant. For each the three task types there are 3! = 6 different possible orders of execution on the three platforms and therefor a total of 18 participants are needed for three task types. All of the participants are unpaid and untrained volunteers, consisting of master students and PhD's at the TU Delft. The experiments will be conducted on location in a meeting room, with one participant at a time.

On each platform a short on-boarding tutorials as discussed in Chapter 4 will have to be executed to ensure the participant has an understanding of the platform's controls.

5.1.2. Evaluation metrics

The experiment will be evaluated by a range of different metrics. These metrics will be discussed in the following subsections.

Accuracy

One of the goals for crowd-mapping is to collect geo-location data for urban objects. Therefor, it is important that the task output is accurate on each platform. However, given that the controls differ on the three platforms, it is possible that the accuracy also differs. To evaluate this, the collected geo-locations of the urban objects are compared to a ground truth dataset. A higher deviation from this dataset indicates a lower accuracy. Similarly the height estimations need to be compared to a ground truth.

Finally, as the users are asked to provide labels at the fix tasks, the accuracy of these labels will be measured. This is done by the verification from workers at the verify tasks, but as this may also include errors they are all manually checked.

Execution time

The time (in seconds) it takes the participant to complete a single find, fix or verify task. This is measured from the time the task is started until the task result is submitted. This does not include the time used to complete the tutorial or filling out the post-task surveys.

Satisfaction

This metric should quantify how satisfied the participant was with executing the tasks on their respective platform. This will be done by asking the participant to provide a rating ranging from 1 to 10 after the experiment, with 1 indicating being very unsatisfied and 10 indicating very satisfied. They will also be able to give additional comments on their experience.

Engagement

A platform has added value when its engagement is high, as that will make people want to come back and continue executing tasks. To make user engagement quantifiable, participants will be asked to fill out a User Engagement Scale (UES) short form. This form contains twelve questions looking at four different factors, as described by O'Brien et al. [26]:

- Focused attention, feeling absorbed in the interaction and losing track of time;
- Perceived usability, negative affect experienced as a result of the interaction and the degree of control and effort expended;
- Aesthetic appeal, the attractiveness and visual appeal of the interface;
- Reward factor, to which extend is the interaction considered rewarding.

Participants will answer questions for these factors on a scale from 1 to 7, after which an engagement score can be calculated by adding all items together and dividing by twelve. The UES survey as filled in by the participants can be seen in Figure 5.1.

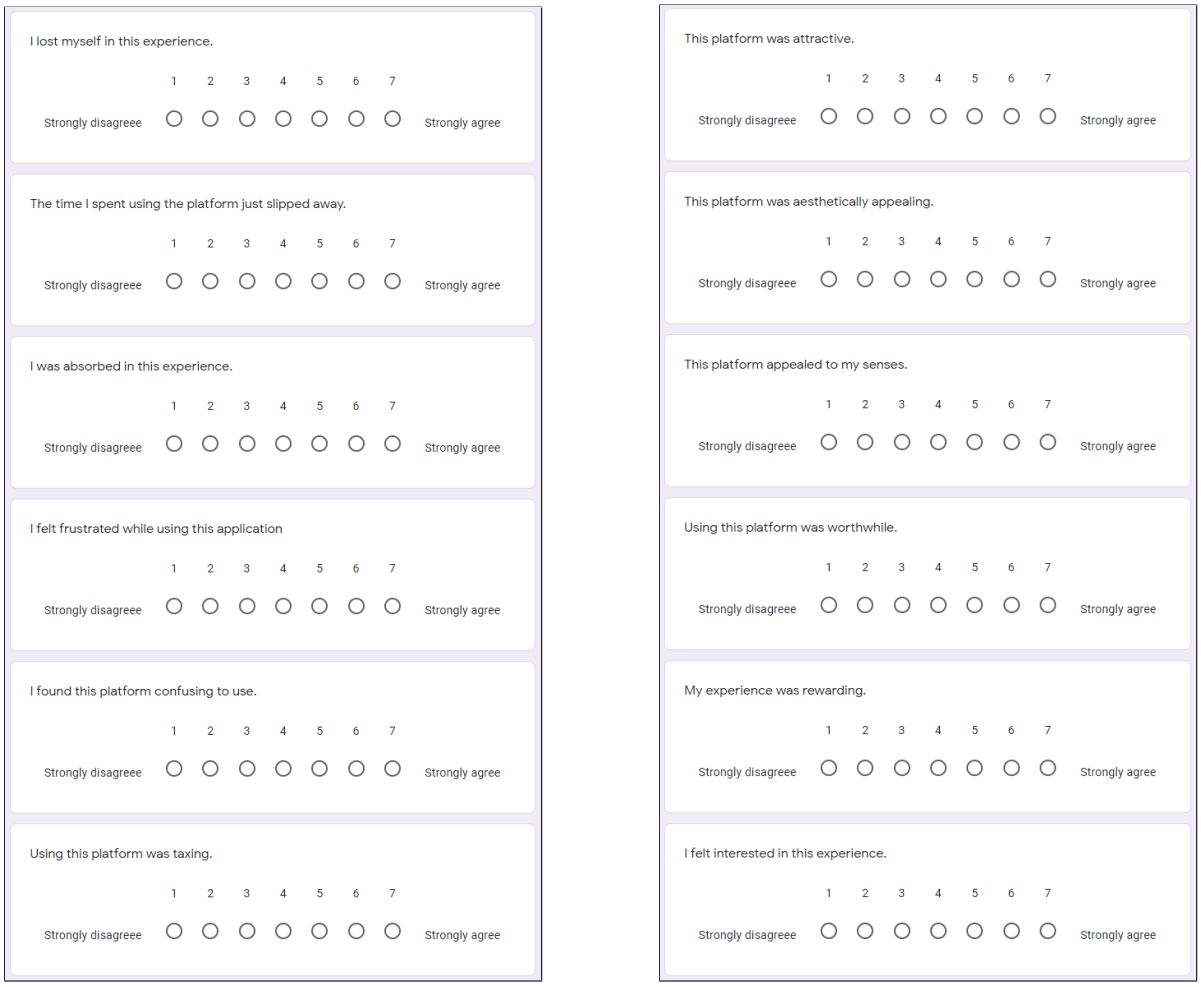


Figure 5.1: UES short form.

Cognitive load

To determine the perceived workload for a certain task type on a certain platform, participants will fill out the NASA Task Load Index (TLX) form. This will for example give insight into how mentally demanding the tasks are. It covers the subjective evaluations of 6 workload-related factors [18]:

- Mental demand: How much mental and perceptual activity was required (e.g., thinking, deciding, calculating, remembering, looking, searching, etc.)? Was the task easy or demanding, simple or complex, exacting or forgiving?
- Physical demand: How much physical activity was required (e.g.. pushing, pulling, turning, controlling, activating, etc.)? Was the task easy or demanding, slow or brisk, slack or strenuous, restful or laborious?
- Temporal demand: How much time pressure did you feel due to the rate or pace at which the tasks or task elements occurred? Was the pace slow and leisurely or rapid and frantic?
- Performance: How successful do you think you were in accomplishing the goals of the task set by the experimenter (or yourself)? How satisfied were you with your performance in accomplishing these goals?
- Effort: How hard did you have to work (mentally and physically) to accomplish your level of performance?
- Frustration level: How insecure, discouraged, irritated, stressed and annoyed versus secure, gratified, content, relaxed and complacent did you feel during the task?

These questions need to be answered on scale of 0 to 100, with 5-point increments (so 20 possible outcomes). An optional second step is the weighing phase, where factors are weighed against each other but as this is rather time consuming this is omitted. This modification to TLX is also referred to as "Raw TLX" [17]. The task load index score (which is on a scale from 0 to 100) is then calculated by taking the sum of all the factors and averaging by dividing by 6, where a lower score indicates a lower perceived cognitive load. The layout of the form can be seen in Figure 5.2.

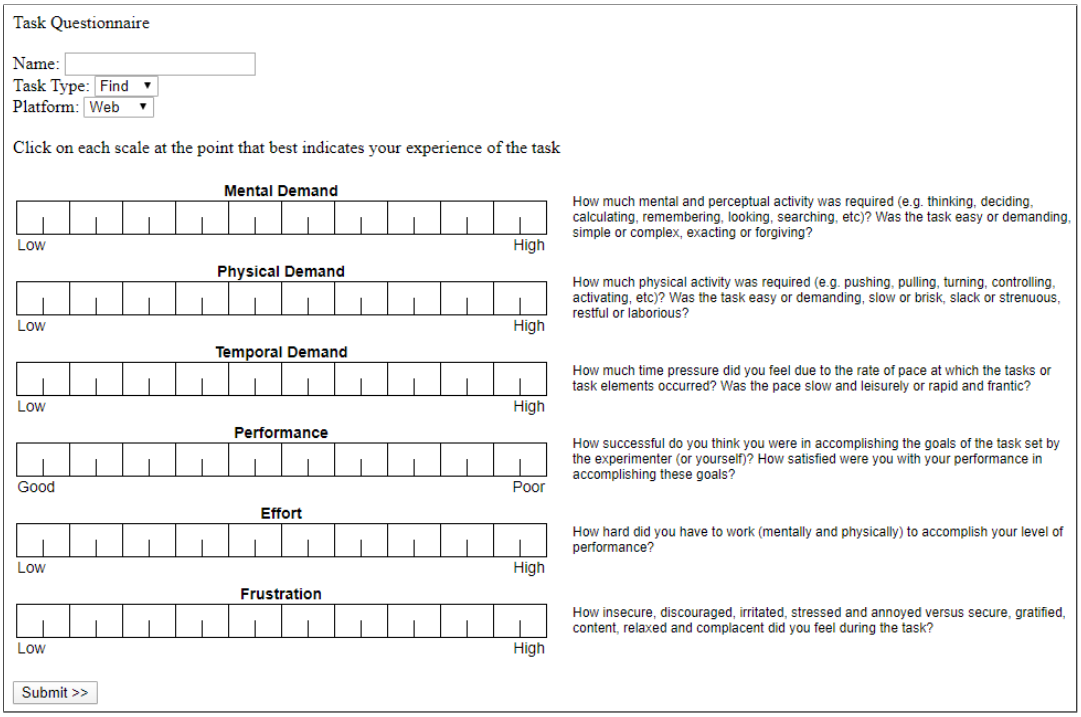


Figure 5.2: NASA TLX form.

5.1.3. Case study: lamp posts

Lamp posts form an integral part of a cityscape. In a large city like for example Amsterdam, thousands of lamp posts are spread around the city. This makes it a good candidate for the experiment as it allows the participants to mark many objects as they are easily found on the street level imagery.

For the find task, each participant will start at a location in Amsterdam and will be asked to find five lamp posts in the area. The municipality of Amsterdam and the government provide a wide range of data about the

city[1]. One of those datasets is the geo-location of lamp posts in the city center containing 240.750 records. This dataset will serve as the ground truth as explained in Section 5.1.2. After finding five objects, they are asked to do the same at two other locations in Amsterdam, giving a total of 15 marked objects per participant for each platform.

Similarly, for the fix task, they are asked to enrich the lamp post annotations done by workers from the find task experiments, by adding labels on for example the state and surroundings of the lamp post and drawing a bounding box around the marked object.

Finally, the verify task participants will be asked to verify the lamp post annotations, where they have to indicate whether the object indeed is a lamp post, whether the bounding box is properly marks the object and whether the labels are correct.

The provided dateset does not contain height information however, so a separate measurement will be done to collect height information. There are many different types of lamp posts in Amsterdam with varying dimensions. Therefor a collection of twenty of a single type of lamp posts with a known height will be manually orchestrated using find tasks on each platform and subsequently annotated in a fix task. This will result in twenty estimated heights for each platform which can be compared to the known height of the lamp post. The type of lantern collected is the traditional Amsterdam "crown lantern" as seen in Figure 5.3, which has a height of 3.75 meters [19].

Figure 5.3: Traditional Amsterdam crown lantern. Source: Google Street View.

5.2. Summary

A qualitative experiment will be executed to evaluate the system. In total 18 participants will be required who will each execute one task type on each of the platforms. These participants are unpaid and untrained volunteers, consisting of master students and PhD's at the TU Delft. The experiments will be conducted on location in a meeting room with one participant at a time. A range of metrics are composed which will be used to evaluate the system. These metrics are accuracy, execution time, satisfaction, engagement and cognitive load. The case study will be focused on lamp posts in Amsterdam, as they are very common in a large city so there will be no issues finding them. Furthermore Amsterdam provides a large dataset containing all the lamp posts in their city with the corresponding geo-locations, which will act as the ground truth for the experiments. As this dataset does not contain height information, a separate test will be done for a particular type of lamp post with a known height.

[1] https://data.amsterdam.nl/, https://data.overheid.nl/

6

Evaluation & discussion

The conducted experiments yielded a range of results which will be discussed in this chapter. Each of the metrics as defined in the previous chapter will be discussed separately in the following sections. Finally it is explored which implications these results have and which limitations there were in this experiment.

6.1. Experiment results

6.1.1. Accuracy

The accuracy of the system was measured for three different components: geo-location estimation accuracy, data enrichment accuracy and verification accuracy. Each of these components are discussed below.

Geo-location estimation

One of the main goals for crowd-mapping is collecting geo-locations. This data which is estimated by the system based on the input of the workers and therefor it is important that the output is reliable. Each of the collected geo-locations during the experiments were compared to the ground truth dataset of street lighting in Amsterdam. This dataset is scanned for the lamp post closest to the crowd-mapped geo-location, by calculating the distance (in meters) between the points in the dataset and the latitude and longitude of the object.

In total 251 geo-locations were collected across the platforms. A boxplot which shows the accuracy is presented in Figure 6.1. All objects with a distance greater than 20 meters from the ground truth were considered invalid and omitted from the data (a total of 9 results across all platforms). This shows the web platform clearly has the least spread an the lowest mean, followed by the mobile platform and finally the VR platform with means/standard deviation of 1.85 ± 2.08, 4.61 ± 4.64 and 6.58 ± 4.87 respectively. For the web platform, 83% of the collected geo-locations were within 2.5 meters of the object in the ground truth. For each of the combination of platforms the difference in distance from the ground truth is of statistical significance (Wilcoxon rank-sum test, $p \leq 0.001$).

Height estimation

For each platform, twenty lamp posts of the same type were found and annotated in a fix task. With the estimated height for each task, the difference was calculated from the ground truth of 3.75 meters. These results are plotted in Figure 6.2. The web platform achieved a mean difference of 0.15 ± 0.12 meters, the mobile platform 0.59 ± 0.26 meters and the VR platform 1.16 ± 0.70 meters. 85% of the estimations on the web platform were within 0.3 meters of the ground truth and all of them were under 0.5 meters, with some results approaching the ground truth to up to 0.009 meters (90 millimeters). As with the geo-location estimation, the mobile and VR show more spread in the accuracy. As the estimated geo-location is a part of the equation for the height estimation, this is to be expected. For each of the combination of platforms the difference from the ground truth is of statistical significance (Wilcoxon rank-sum test, $p \leq 0.006$).

Data enrichment

For the fix task, participants had to enrich the objects found by the find task participants. Here they have to draw bounding boxes and add relevant labels. The accuracy of these actions are presented in Table 6.1. Here

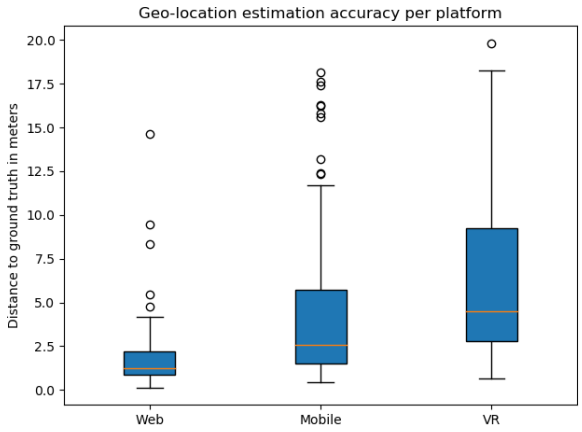


Figure 6.1: Geo-location estimation accuracy per platform.

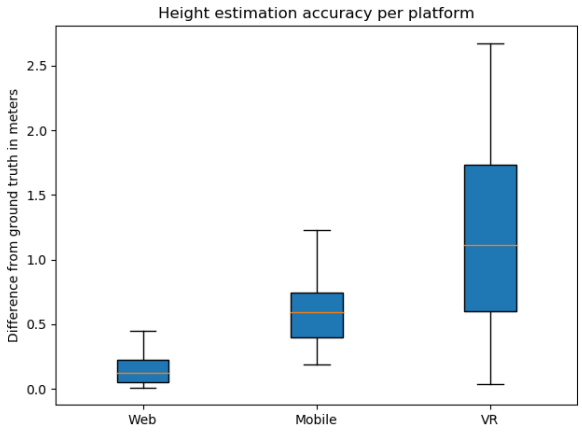


Figure 6.2: Height estimation accuracy per platform.

it becomes clear that the VR platform performs best for drawing bounding boxes with an accuracy of 88.4%. Most bounding boxes errors among all platforms were caused by not containing the entire object, but only a part of it. It appears the VR participants were able to be more accurate in drawing the bounding boxes. The label accuracy was roughly similar for the web and mobile platforms, but was worse on the VR platform. It should also be noted that many participants only chose one or more of the suggested labels as opposed to adding new labels. It requires less effort to select a predefined label and therefor the users preferred to do this, but as they rarely included any new labels, the total set of labels remained small.

	Web	**Mobile**	**VR**
% Correct bounding boxes	72.9	71.4	**88.4**
% Correct labels	**98.9**	96.2	83.6

Table 6.1: Accuracy of fix task output for each platform and task type.

Verification

As a final step, the find and fix outputs were verified by the experiment participants. First they check whether the marked object complies with the task description. Then they check whether the bounding box properly contains the object and if the added labels are relevant to the object and it's surroundings. The results of these experiments can be seen in Table 6.2.

All platforms performed well in determining whether the object matched the description, with a slight edge for the web platform. The platforms also performed similarly at verifying the bounding boxes, with the VR platform marginally outperforming the others. Most errors were made by users viewing a bounding box

as correct when a part of the object was not included in the box. And finally for the label verification, the platforms again performed similarly.

	Web	**Mobile**	**VR**
% Correct object verification	**97.1**	92.9	90.5
% Correct bounding boxes verification	70.0	78.6	**81.0**
% Correct label verification	**90.5**	88.9	84.5

Table 6.2: Accuracy of verify task output for each platform and task type.

6.1.2. Execution time

For each of the tasks, the execution time was measured when the participants were conducting the experiments. For the find task this the time it takes to find a single object: the timer starts when first starting the task and stops when an object is found and marked. The timer is then started again and stopped when another object is found and so on. With the fix and verify task, it is the time it takes to complete a single task. The average execution times are shown in Table 6.3.

The mobile platform had the shortest execution time for every task type with the lowest spread. Similarly, it was observed that the VR platform has the longest execution times for all task types. The obvious outlier is the VR fix task execution time, but here it should also be noted that it did output the most accurate bounding boxes, as seen in Table 6.1.

	Web	**Mobile**	**VR**
Find	18 ± 15	$\mathbf{16 \pm 15}$	20 ± 19
Fix	25 ± 21	$\mathbf{19 \pm 13}$	41 ± 36
Verify	17 ± 17	$\mathbf{13 \pm 9}$	27 ± 19

Table 6.3: Average execution time in seconds for each platform and task type.

6.1.3. Engagement

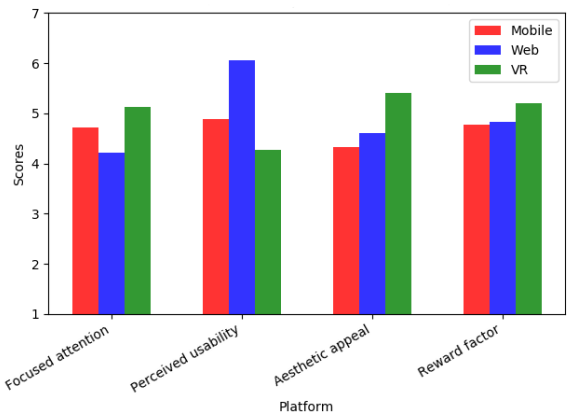
For all task types and platform, a user engagement score (UES) was calculated, based on the responses participants gave for the UES short-form. This is a score on a scale of 1 to 7, with a higher score meaning the participants were more engaged while performing the tasks. The respective score are presented in Table 6.4. The scores for each platform and the different UES factors can be seen in Figure 6.3.

Project Sidewalk mentions that engagement remains a challenge. 63% of paid workers completed at least one mission, while only 19.1% of volunteers did the same [31, 32]. Therefor, the findings from this experiment could be of use for increasing the engagement.

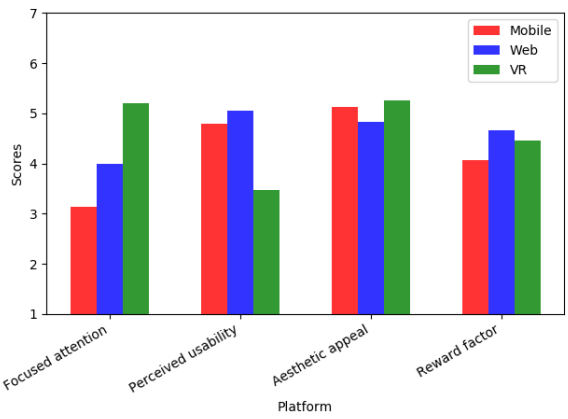
The user engagement scores are relatively similar for the different platforms and task types, except for the mobile platform, which got both the lowest and the highest score by some margin. With the fix task, the frustration factor was higher for the mobile participants as the sometimes were not able to be as accurate as they wanted to be. For the verify task, all mobile platform participants responded with lowest score for confusion, indicating that the it was very clear for them what was expected from them. Additionally, most participants indicated that time slipped away while executing the task, making them more willing to continue executing the tasks.

	Web	**Mobile**	**VR**
Find	4.93 ± 0.60	4.68 ± 0.45	$\mathbf{5.00 \pm 0.50}$
Fix	$\mathbf{4.64 \pm 0.65}$	4.28 ± 1.01	4.60 ± 0.28
Verify	4.72 ± 0.74	$\mathbf{5.26 \pm 1.12}$	4.29 ± 1.16

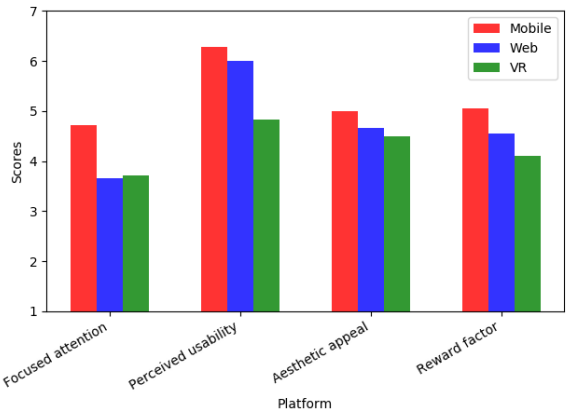
Table 6.4: User engagement score for each platform and task type.



(a) UES-SF score by platform for find task



(b) UES-SF score by platform for fix task



(c) UES-SF score by platform for verify task

Figure 6.3: Average score per UES factor for each platform and task type.

6.1.4. Cognitive load

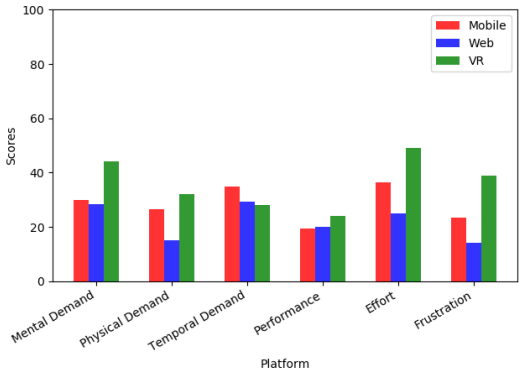
Each participant filled in a "raw" Task Load Index form, to measure the cognitive load for each platform and task type. Based on the responses, the average Task Load Index scores could be calculated, where a lower score means a lower cognitive load. These averages are presented in Table 6.5 and the individual factors making up these scores are plotted in Figure 6.4.

The VR platform has by far the highest scores across the different task types. Most participants experienced it as a much more intensive experience, as it requires more physical interaction and is significantly more immersive, yielding higher Mental Demand and Physical Demand scores. The frustration factor was also much higher for the VR platform as participants experienced a steeper learning curve.

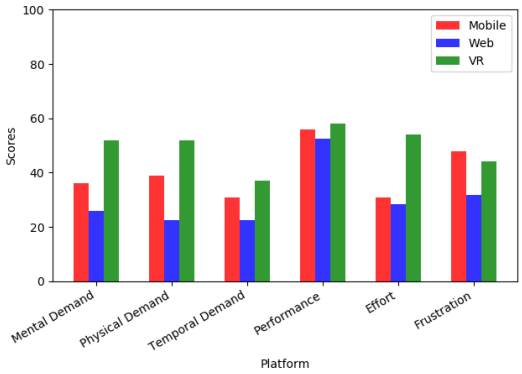
For the find task, the web platform has the lowest TLX score, as users found it intuitive to navigate through the streets using the mouse controls. Similarly for the fix task, the web platform also has the lowest score as participants indicated that it required less effort to draw the bounding boxes, which caused significantly less frustration. And finally for the verify task, participants for the mobile and web platforms observed low Temporal Demand, indicating that they did not experience much time pressure while executing the task. The mobile platform has the lowest TLX score here, as users found it intuitive to navigate using the touch controls.

	Web	**Mobile**	**VR**
Find	**21.9 ± 12.39**	28.5 ± 16.50	36.0 ± 8.14
Fix	**30.6 ± 13.66**	40.2 ± 5.88	49.5 ± 11.29
Verify	19.9 ± 13.59	**16.7 ± 13.97**	33.6 ± 16.26

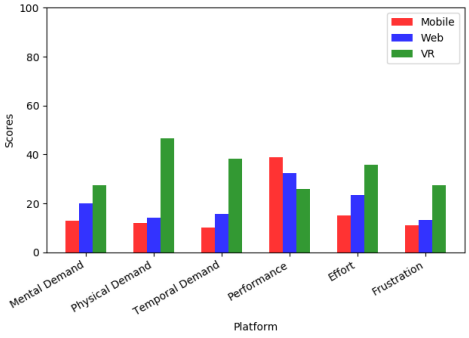
Table 6.5: Task Load Index score for each platform and task type.



(a) Average TLX factor scores by platform for find task



(b) Average TLX factor scores by platform for fix task



(c) Average TLX factor scores by platform for verify task

Figure 6.4: Average score per TLX factor for each platform and task type.

6.1.5. Satisfaction

After each experiment on a certain platform, users were asked to give a general rating on their experience, along with additional comments. The average of these ratings is found in Table 6.6.

For the find task, the web platform marginally beat the mobile and the VR platform. Some users preferred using a mouse to mark the objects, whereas others preferred the touch input of the mobile platform. This score is slightly lower than the findings by Qiu, where users gave the task a satisfaction score of 3.9/5 (or 7.8/10) [27].

With the fix task, the web platform again has the highest average rating. Users found the web platform more convenient to draw the bounding boxes, as it is easier to be more precise with a mouse.

However, for the verify task the mobile platform performed significantly better. Users found the touch input a convenient and quick way of interacting with the application (which also reflected in the lower TLX score) and could see themselves performing this type of task for example when they have some downtime. The most divide in ratings in general was observed for the VR platform. Participants with no prior VR experience found it harder to get into, whereas participants who had experienced VR before were more comfortable using the platform.

	Web	Mobile	VR
Find	**7.0 ± 1.15**	6.8 ± 0.69	6.8 ± 0.75
Fix	**8.0 ± 1.15**	7.0 ± 0.63	7.2 ± 0.75
Verify	7.7 ± 1.37	**8.8 ± 1.21**	7.0 ± 2.65

Table 6.6: Average user rating for each platform and task type

6.2. Discussion

In this section, it is discussed which implications can be made based on the collected data. Furthermore, there were some limitations in this experiment which will also be discussed in this section.

6.2.1. Implications

The experiments showed that the location estimation method using two angles is able to achieve accurate results when the platform supports an input method which allows the user to mark the object with precision, as is the case with the mouse on the web platform. It is much easier to accurately click on a specific location with a mouse than using your finger. With mobile VR the workers have to select the location using their head movement and as it's hard to keep your head completely still it will complicate being accurate. Especially considering the fact that with the location estimation algorithm used (as explained in Section 3.5.1), a worker has to mark the same object twice, from different angles, which further increases the chance of being inaccurate. Accuracy suffers when the marked position slightly differs for the two angles. A solution could be to let the workers mark more than two angles. However, this would significantly increase execution time for the find task. With 83% of the estimated geo-locations being within 2.5 meters of the ground truth, the web platform in combination with the two angle location estimation method can be considered sufficiently accurate. The other platform also performed reasonable, with the majority of the results being within 5 meters of the ground truth. Depending on the required accuracy, all platforms could be deployed to execute find tasks.

The height estimation showed very accurate results for the web platform, with 85% of the estimation being within 30 centimeters of the ground truth. The mobile platform was less accurate, with a mean difference of 59 centimeters, followed by the VR platform with a mean of 116 centimeters. As explained in Section 3.5.2, part of the estimation is calculating the distance from the camera to the marked object, which requires the estimated geo-location of the object. As a result, the height estimation suffers when the geo-location lacks accuracy. The more accurate the geo-location gets, the more accurate the height estimation will get. For now, it depends on the application if the height estimation could be of use. If a rough indication of height is required, all platforms may supply sufficient estimations, but when more accuracy is required the web platform should be used.

The data enrichment part of the experiment showed mixed results, with the VR platform being significantly more accurate in drawing bounding boxes, but performing worse at generating relevant labels. The bounding box accuracy could be a result of the fact that they are closer to the screen and therefor are better able to make out the object. The VR label generation accuracy could be caused by errors from the voice recognition. Most bounding box errors among the platforms were made by participants not correctly following the instructions on how to draw a bounding box, so this could be improved upon. Additionally, the set of text labels did not grow much after the initial workers added labels, as these labels are presented as suggestions for other workers. As a result they often only picked the predefined labels as opposed to adding new labels.

Verification accuracy was similar across all platforms. Ideally the bounding box verification accuracy would be higher, which could be achieved by having more workers verify the same data, similar to for example Project Sidewalk [32].

The mobile platform proved to be the quickest for all tasks in terms of execution time, which might be because with touch control, every element that can be interacted with on the screen is within finger's reach, making the interactions quicker. The VR platform was the slowest with a fix task execution time average of 41 seconds. Navigation and interaction is slower on this platform as the ways of interacting with the application are limited. As noted before the bounding box accuracy was much higher than the other platforms, so this seems to be compromised by the execution time.

User engagement and cognitive load were other factors that were measured during the experiments. The VR platform received significantly higher TLX scores given the more immersive and physically demanding nature of the platform. This did not appear to heavily impact the user engagement however, with the exception of the verify task, where the immersion of VR did not seem to have any added value and users were less engaged.

All results considered, the platforms in their current implementation all appear to have a specific task type at which it performs best overall. This is also roughly reflected in the user satisfaction scores. The web platform seems best suited for the find tasks, with the high accuracy, reasonable execution time, high engagement score and lowest TLX score. Similarly the VR platform performs well at the fix tasks, although the execution time and perceived task load have to be considered when utilizing the platform. Finally, the mobile platform seems well suited for verify task execution. Participant executed the tasks very quickly, whilst still being reasonably accurate, receiving high user engagement scores and low TLX scores.

6.2.2. Limitations

The main limitation of the experiment was the limited amount of participants. Mainly the engagement, cognitive load and satisfaction metrics suffer from this as this is subjective data which would benefit from having more input. Similarly it could benefit from a more diverse demographic, as all participants were affiliated with the TU Delft.

The results of these experiments come from a single case study with lamp posts. Further research needs to be done with different objects to determine whether the type of object influences the metrics. Additionally, no ground truth data set was available for the height estimation, so this was manually orchestrated. In the interest of time, the size of this dataset was limited to twenty lamp posts.

7

Conclusions

In this thesis it was studied how to design and implement a multi-platform crowd-mapping application for urban object mapping using street-level imagery. In the process novel techniques were researched and applied, such as mobile virtual reality, location estimation and height estimation. The resulting application showed promising results regarding location/height estimation accuracy and user satisfaction. During this research, the four research questions that were posed in this thesis were answered.

To answer **RQ1**, a literature study was conducted. From this study it became evident that the topic of crowd-mapping is a relatively new field of research. The origins can be found in managing crises and disasters in 2008 when the Ushahidi system was developed to map post-election violence in Kenya. In 2010 this platform was used to collect data about events surrounding the Haiti earthquake. In the following years new crowd-mapping systems would emerge and in 2011 the concept was defined as "combining the aggregation of a Geographic Information System and crowd-generated content" [8].

Based on the nine systems found in the studied literature, a taxonomy was made for crowd-mapping using street-level imagery. These systems all map properties, which are either subjective or objective, using a certain strategy. This information formed a solid basis for this thesis, but a research gap was recognized. First of all, it must be stated that crowd-mapping systems still have added value in a world of computer vision systems, as these systems tend to output false-positives which would be avoided when judged by humans. Furthermore these algorithms require human-annotated training data. All of the existing crowd-mapping platforms were developed on web-based platforms. This thesis serves as a first step to research alternative approaches by implementing the mobile application and the mobile VR platform. Similarly as mentioned earlier, the systems tend to have a main focus on either accessibility data or cityscape data, but none of them abstract the idea of crowd-mapping in a manner that the system can be used for general purpose goals. Finally, geo-location estimation remained a challenge throughout the systems and none of them estimated the height of the mapped objects. This thesis proposes novel approaches to tackle these challenges.

As part of **RQ2** it was researched how to abstract the concept of crowd-mapping and how to develop a platform with which task requesters can deploy general purpose crowd-mapping tasks. A general task requester workflow was composed which consists of specifying task details, deploying the task and analyzing the results. Based on this workflow, a crowd-mapping dashboard was developed where a task can be created by providing the task goal, budget and area bounds for the task. After deploying the task, it will be instantly available for workers on all task executing platforms using the developed API. After they submit their results they are aggregated an presented in the dashboard as part of the analysis section. The results are shown on an interactive map with a heatmap which shows the hot-spots of objects in the task area.

As the aim of **RQ3** was to research time- and cost-efficiently task execution with high accuracy, it was decided to use the find-fix-verify pattern. This is a proven strategy in literature and splits tasks into a series of generation and review stages [7]. When a task is deployed, find tasks are generated first. In these tasks the workers are free to virtually roam around a designated area, with the goal of finding objects matching the task description as provided by the requester. The aim here is to collect geo-location information for these objects, which has to be estimated from the workers input. A novel approach was proposed which involves marking the object from two angles and calculating the intersection of the lines figuratively drawn from both angles.

The second stage is the fix task, with the aim of enriching the data generated in the find task, or discarding it when the data is not accurate. The enrichment part consists of drawing a bounding box over the object and providing labels which are relevant to the object or it's surroundings. This data can be used to train a computer vision system and the bounding box is also used to calculate the height of an object. This is done by calculating the angle from the camera to the highest point of the bounding box and using the distance from the camera to the object. This data can be used to calculate the height by simply calculating the side of triangle given a known side (the distance between camera and object) and an angle.

The final stage of the task execution process is the verify task. As the name suggests, the aim of this task is to verify the input which was provided by other workers. They do so by indicating whether the marked object matches the task description and whether the bounding box properly encloses the object. Furthermore they judge whether the provided labels are relevant for the object and it's surroundings.

As this thesis uses novel approaches for crowd-mapping, it was evaluated how the different platforms affect the worker satisfaction, perceived cognitive load, engagement, execution time and output quality as part of **RQ4**. As a general note, the input method did seem to affect the accuracy of the output. With the find task for example it is more challenging to accurately mark an object using touch-based or gaze-based controls, which should be a consideration when utilizing the two-angle location estimation approach.

The execution time for each of the tasks and platforms was measured during the experimentation. This showed that the mobile platform consistently had the lowest execution time for all tasks. All task types and platforms had an average of well below one minute however. The mobile VR platform tasks took the longest to execute on average as the gaze based interaction tends to be slower than touch or mouse input.

An important part of the experimentation was the task output quality and accuracy. The proposed geo-location technique showed promising results, especially from the web platform with a mean difference of 1.85 meters from the ground truth and 83% within 2.5 meters. The web platform also performed very well at height estimation with a mean difference of 15 centimeters and 85% of the estimations being within 30 centimeters of the ground truth. As the height estimation technique uses the result of the geo-location estimation, these results can be improved by improving the geo-location estimation technique.

The bounding box drawing was relatively accurate with 71.4% and higher of the bounding boxes being drawn correctly. Most bounding box errors were made by users not correctly following the instructions of precisely enclosing just the object and not more or less than that. Making these instructions more obvious could affect the results.

An interesting finding was that suggesting labels which have been entered before by other workers should be done with caution. Workers tend to choose the suggested labels more frequently which limits the amount of new labels that are generated. The label accuracy was very accurate with a percentage of correct labels of 83.6% for the mobile VR platform and the web and mobile platforms approaching 100%. This could however also be caused by the workers choosing the suggested labels, which keeps the total set of different labels limited.

The final part of accuracy measurements was done for the verification part of the system. The accuracy was close to or over 90% for the label and object verification. The bounding box verification accuracy suffered from the same issue as the bounding box drawing, with people not strictly following the instructions.

Experimentation suggested that the platforms in their current implementation all have a specific task type which produces the highest output quality and satisfaction. The web platform produces high-quality geo-location estimations whilst having reasonable execution times, high engagement scores and low perceived cognitive load for the find tasks. Lower label accuracy aside, the mobile platform performs well at the fix task with the bounding box accuracy outperforming that of the web and mobile platforms with a significant margin. Finally the mobile platform seems best suited for the verify tasks with the combination of high accuracy, low execution time, high user engagement scores and low perceived cognitive load.

The perceived cognitive load was relatively high in general for the mobile VR platform, as users found it harder to learn the basic controls, which lead to a higher frustration factor. The physical part of having to move your head for virtual reality also naturally resulted in higher mental demand and physical demand factors. The average perceived cognitive load was lower for the web and mobile platform. Furthermore, the different platforms seemed to marginally affect the satisfaction ratings but the sample size is too low to draw definitive conclusions on this subject.

7.1. Future work

One of the main limitations of the experiment was the limited amount of participants with a total of 18. More research should be done with additional participants to determine whether this affects the results. These participants should also have more diverse background as opposed to being limited to having a TU Delft background.

The experiment in this work focused on a single case study with lamp posts in Amsterdam. Further research needs to be done with different objects and different cities to determine whether the type of object influences the metrics. Similarly, the only tested geo-location technique is the two-angled approach which was proposed in this thesis. An alternative approach could be to use more than two angles, which might positively affect the output quality but simultaneously increase the execution time. Additional research could be done on geo-location estimation on platforms with less accurate input methods like the mobile and mobile VR platforms used in this thesis.

Some workers had difficulty getting used to the mobile VR platform interaction. This caused frustration and higher perceived cognitive load. More work could be put into improvements of the user interface and interaction methods in order to reduce the frustration factor.

Another subject which was proposed in this thesis is object height estimation. As there was no ground truth data set was available at the time of writing, this was composed manually. In the interest of time, the size of this dataset was limited to twenty lamp posts. A larger ground truth dataset should be found or generated and be compared to the output from crowd-mapping tasks.

Worker selection and task assignment strategies were considered out of the scope of this thesis. The task assignment is currently implemented as a first-come-first-serve strategy. Considering worker quality and reliability could improve the output quality. Furthermore, the task output only went through a single stage of verification, meaning that the task output of a crowd worker is only verified by a single other crowd worker. As a final future work proposal it could be researched whether multiple verification stages would improve output quality and how this affects the execution time.

Bibliography

[1] Designing for google cardboard. https://designguidelines.withgoogle.com/cardboard/. Accessed: 2020-01-20.

[2] Google cardboard – google vr. https://arvr.google.com/cardboard/, . Accessed: 2020-04-29.

[3] Oculus rift s: Vr headset for vr ready pcs | oculus. https://www.oculus.com/rift-s/, . Accessed: 2020-04-29.

[4] Newzoo's global mobile market report: Insights into the world's 3.2 billion smartphone users, the devices they use the mobile games they play. https://newzoo.com/insights/articles/newzoos-global-mobile-market-report-insights-into-the-worlds-3-2-billion-smartphone-users-the-devi Accessed: 2020-01-13.

[5] Virtual reality (vr) - statistics & facts. https://www.statista.com/topics/2532/virtual-reality-vr/. Accessed: 2020-01-13.

[6] Michael DM Bader, Stephen J Mooney, Blake Bennett, and Andrew G Rundle. The promise, practicalities, and perils of virtually auditing neighborhoods using google street view. *The ANNALS of the American Academy of Political and Social Science*, 669(1):18–40, 2017.

[7] Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 313–322. ACM, 2010.

[8] Carlos Caminha and Vasco Furtado. Modeling user reports in crowdmaps as a complex network. In *Proceedings of 21st International World Wide Web Conference. Citeseer*, 2012.

[9] Nathan Eagle. txteagle: Mobile crowdsourcing. In *International Conference on Internationalization, Design and Global Development*, pages 447–456. Springer, 2009.

[10] Raghu K Ganti, Fan Ye, and Hui Lei. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine*, 49(11):32–39, 2011.

[11] Mar Gonzalez-Franco and Jaron Lanier. Model of illusions and virtual reality. *Frontiers in psychology*, 8: 1125, 2017.

[12] Michael F Goodchild. Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69(4):211–221, 2007.

[13] Aakar Gupta, William Thies, Edward Cutrell, and Ravin Balakrishnan. mclerk: enabling mobile crowdsourcing in developing regions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1843–1852, 2012.

[14] Kotaro Hara, Vicki Le, and Jon Froehlich. Combining crowdsourcing and google street view to identify street-level accessibility problems. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 631–640. ACM, 2013.

[15] Kotaro Hara, Jin Sun, Robert Moore, David Jacobs, and Jon Froehlich. Tohme: detecting curb ramps in google street view using crowdsourcing, computer vision, and machine learning. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 189–204. ACM, 2014.

[16] Kotaro Hara, Shiri Azenkot, Megan Campbell, Cynthia L Bennett, Vicki Le, Sean Pannella, Robert Moore, Kelly Minckler, Rochelle H Ng, and Jon E Froehlich. Improving public transit accessibility for blind riders by crowdsourcing bus stop landmark locations with google street view: An extended analysis. *ACM Transactions on Accessible Computing (TACCESS)*, 6(2):5, 2015.

[17] Sandra G Hart. Nasa-task load index (nasa-tlx); 20 years later. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 50, pages 904–908. Sage Publications Sage CA: Los Angeles, CA, 2006.

[18] Sandra G Hart and Lowell E Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, volume 52, pages 139–183. Elsevier, 1988.

[19] Arnold Korporaal. Nieuw licht op oude grachten. 2005.

[20] HARA Kotaro, Victoria Le, and Jon Froehlich. A feasibility study of crowdsourcing and google street view to determine sidewalk accessibility. 2012.

[21] Xiaojiang Li and Carlo Ratti. Mapping the spatial distribution of shade provision of street trees in boston using google street view panoramas. *Urban Forestry & Urban Greening*, 31:109–119, 2018.

[22] Xiaojiang Li, Chuanrong Zhang, Weidong Li, Robert Ricard, Qingyan Meng, and Weixing Zhang. Assessing street-level urban greenery using google street view and a modified green view index. *Urban Forestry & Urban Greening*, 14(3):675–685, 2015.

[23] Xiao Ma, Megan Cackett, Leslie Park, Eric Chien, and Mor Naaman. Web-based vr experiments powered by the crowd. In *Proceedings of the 2018 World Wide Web Conference*, pages 33–43, 2018.

[24] Patrick Meier. Ushahidi as a liberation technology. *Liberation technology: Social media and the struggle for democracy*, pages 95–109, 2012.

[25] FA Mora. Innovating in the midst of crisis: A case study of ushahidi. *Submitted for publication to SAGE Convergence Journal*, 2011.

[26] Heather L O'Brien, Paul Cairns, and Mark Hall. A practical approach to measuring user engagement with the refined user engagement scale (ues) and new ues short form. *International Journal of Human-Computer Studies*, 112:28–39, 2018.

[27] Sihang Qiu, Achilleas Psyllidis, Alessandro Bozzon, and Geert-Jan Houben. Crowd-mapping urban objects from street-level imagery. In *The World Wide Web Conference*, WWW '19, page 1521–1531, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748. doi: 10.1145/3308558.3313651. URL `https://doi.org/10.1145/3308558.3313651`.

[28] Daniele Quercia. Urban: Crowdsourcing for the good of london. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 591–592. ACM, 2013.

[29] Kenneth Rogers and R Scholz. Crowdmapping the classroom with ushahidi. *Learning Through Digital Media Experiments in Technology and Pedagogy*, 2011.

[30] Salvador Ruiz-Correa, Darshan Santani, and Daniel Gatica-Perez. The young and the city: Crowdsourcing urban awareness in a developing country. In *Proceedings of the First International Conference on IoT in Urban Space*, pages 74–79. ICST (Institute for Computer Sciences, Social-Informatics and . . . , 2014.

[31] Manaswi Saha, Kotaro Hara, Soheil Behnezhad, Anthony Li, Michael Saugstad, Hanuma Maddali, Sage Chen, and Jon E Froehlich. A pilot deployment of an online tool for large-scale virtual auditing of urban accessibility. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 305–306. ACM, 2017.

[32] Manaswi Saha, Michael Saugstad, Hanuma Teja Maddali, Aileen Zeng, Ryan Holland, Steven Bower, Aditya Dash, Sage Chen, Anthony Li, Kotaro Hara, and Jon Froehlich. Project sidewalk: A web-based crowdsourcing tool for collecting sidewalk accessibility data at scale project sidewalk: A web-based crowdsourcing tool for collecting sidewalk accessibil-ity data at scale. 05 2019. doi: 10.1145/3290605.3300292.

[33] Philip Salesses, Katja Schechtner, and César A Hidalgo. The collaborative image of the city: mapping the inequality of urban perception. *PloS one*, 8(7):e68400, 2013.

[34] Abdul Rehman Shahid and Amany Elbanna. The impact of crowdsourcing on organisational practices: The case of crowdmapping. 2015.

[35] Mel Slater. Place illusion and plausibility can lead to realistic behaviour in immersive virtual environments. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1535):3549–3557, 2009.

[36] Ed Tobias. Using twitter and other social media platforms to provide situational awareness during an incident. *Journal of business continuity & emergency planning*, 5(3):208–223, 2011.

[37] Jan D Wegner, Steven Branson, David Hall, Konrad Schindler, and Pietro Perona. Cataloging public objects using aerial and street-level images-urban trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6014–6023, 2016.

[38] Tingxin Yan, Matt Marzilli, Ryan Holmes, Deepak Ganesan, and Mark Corner. mcrowd: a platform for mobile crowdsourcing. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pages 347–348, 2009.