IT UNIVERSITY OF COPENHAGEN

Master Thesis

Geodata-Guided Defrosting Robot: Image Segmentation for Path Planning

By Malthe Mejdal Nielsen

Submission date: 03/06/2024

Course code: KISPECI1SE

Programme: Msc. Software Design

Student: Malthe Mejdal Nielsen

Student ID: MEMN@ITU.DK

Supervisor: Kasper Støy

Abstract

Defrosting sidewalks is crucial in regions experiencing sub-zero temperatures and precipitation, yet it is a time-consuming and expensive task with significant environmental impact. A robotic solution is developed to defrost autonomously, by a company aiming to use segmented geographical data for path planning. Segmentation of sidewalks is a shallowly researched area, that this thesis seeks to advance by answering the following research question:

How can geodata be utilized to generate segmented maps of public sidewalks suitable for path planning?

The research is conducted by analyzing data sources and evaluating an available orthophoto segmentation model. A custom segmentation model is developed using architectural maps, achieving higher precision than the orthophoto segmenting model. The custom model segments sidewalks with an average precision of 35%, recall of 68%, and accuracy of 89%. It is determined how to convert the maps, and successfully run path planning using a simple algorithm. Though the research successfully utilizes geographical data to segment maps suitable for path planning, sources of errors are determined that are subject to future work for optimization.

Acknowledgement

I would like to thank Capra Robotics for providing their problem statement, serving as the foundation for this research. I would like to express my gratitude to Professor Kasper Støy for his supervision and contribution of knowledge within the fields of AI and research. Special thanks to the following people and organizations for contributing with knowledge of data sources: Technical designer Birthe Nielsen, Dataforsyningen, and The Danish Geodata Agency.

I appreciate the guidance from special advisor Lottie Rosamund Greenwood of IT University of Copenhagen and teaching assistant Lasse Thamsen of Aarhus University on running models on a high-performance cluster, an area in which I have no prior experience.

Declaration of plagiarism

I hereby declare that this thesis, entitled "Geodata-Guided Defrosting Robot: Image Segmentation for Path Planning" represents my own work and findings during the designated project period and has not been submitted elsewhere. All sources of information and code have been acknowledged. Generative AI has been used for troubleshooting. The thesis does not contain material generated by such models.

Malthe Mejdal Nielsen 3rd June 2024

Contents

List of Figures

List of Tables

Software

All relevant code and data are available from the following ITU GitHub repository with access limited to IT University of Copenhagen:
www.Github.itu.dk/memn/MasterThesis

The content is also temporarily available from the following public GitHub repository:
www.Github.com/MaltheMejdal/MasterThesis

1. Introduction

Defrosting is crucial in regions experiencing sub-zero-degree temperatures and precipitation, posing challenges regarding safety, finances, legal, and the environment.

The primary method used for defrosting is spreading large amounts of salt to lower the freezing point of water, causing ice to melt. Icy surfaces pose significant safety risks as the friction is reduced and causes accidents. This is documented by 92 percent of all slippery accidents reported to the insurance company Almindelig Brand in 2021 occurring in the cold months of December, January, and February [1].

Lack of defrosting not only endangers people's health but can also carry legal consequences for those responsible for the roads, cycling lanes, sidewalks, and walking paths. The municipally is responsible for clearing and defrosting any public area, while private property owners are responsible for sidewalks bounding the property parameter. Property owners are limited to defrosting a maximum length of 10 meters [2]. Additionally, property owners are entirely responsible for privately owned roads and sidewalks. While defrosting of public sidewalks is a shared responsibility between the municipally and property owners, it is a task that isn't always fulfilled. Defrosting is time-consuming and costly to outsource to contractors.

The expenses of salting have increased significantly [3], due to a lack of drivers and contractors. Furthermore, the demand for salt tends to increase during excessively cold winters, resulting in a shortage and increased prices.

It is estimated that 66 million tons of road salt are used in Denmark annually [4], which tears on vehicles, roads, sidewalks, and underground pipes. The salt is eventually dissolved with the risk of contaminating groundwater and ecosystems.

There are multiple initiatives towards optimizing pedestrian safety and reducing the financial and environmental impact of defrosting. One of the approaches is to use robots spreading an environmentally optimized brine precisely.

1.1. Geodata-Guided Defrosting Robot

The Aarhus-based company Capra Robotics is looking to optimize maintenance and service operations in urban areas focusing on the "Smart City" sector. The concepts utilize their universal robot chassis and custom tools to remove cigarette buds and chewing gum, and to defrost sidewalks and walking paths. Capra robotics uses brine for defrosting, resulting in 30-70 percent less salt consumption [4]. The brine is applied using a spraying tool seen in figure 1.1 mounted on the robotic platform. This robot configuration is named Frosty.

Figure 1.1: Capra Robotics platform with Saltnex spreader. [5]

Frosty is currently a manually operated prototype but Capra Robotics is aiming to offer an autonomous solution to reduce the cost of operation. The robot will cover public sidewalks and public walkable areas while excluding bicycle paths, roads, stairs, and vegetated areas. The robot will use pedestrian crossings to move between areas but not defrost crossing sections located on roads.

As Capra is looking to offer their services for large areas with multiple refilling stations, manual path planning will be time-consuming and complex. Autonomously generated paths require accurate maps of the areas, which may be generated from geodata.

1.2. Research Question

This research seeks to discover:
How can geodata be utilized to generate segmented maps of public sidewalks suitable for path planning?

The following subjects are researched to answer the research question.

- What data is available and can be used for segmentation?
- Which segmentation models are available?
- How do the segmentation models perform?
- How should a custom model be constructed and how does it perform?
- How can the segmented maps be prepared for path planning?

1.3. Reading Guide

The following guide gives an insight into the content of each chapter.

Chapter 2 - Geographic Data Sourcing

The available map services of geographic data and specifications of APIs are explored. The impact of coordinate systems, tile systems, and projections is investigated.

Chapter 3 - Ortophoto Segmentation

The relevant available segmentation models are reviewed and two models are implemented. Segment Anything Model (SAM) is discarded due to low performance and Tile2Net is further tested. Data sources for Tile2Net are evaluated. The model is tested on orthophoto data from ten locations in Aarhus and performance metrics are evaluated. Suggestions are made for optimization.

Chapter 4 - Architectural Map Segmentation

Architectural maps are chosen from map services and a segmentation model is constructed based on OpenCV-offered image processing methods. The model is tested on ten locations in Aarhus and performance metrics are evaluated. Suggestions are made for optimization.

Chapter 5 - Segmentation Model Comparison

Tile2Net and Architectural Map Segmentation (AMS) results are compared and analyzed.

Chapter 6 - Map Suitability for Path Planning

Segmented maps are converted to a lower resolution of nodes for path planning, using a simple path planning algorithm to test the suitability of the maps. Results are evaluated and suggestions are made for optimization to enable real-world usage.

Chapter 7 - Discussion

The results are discussed, and the research outcome is compared to the research question. Possible sources of errors and model behavior are discussed and suggestions for optimization are presented.

Chapter 8 - Conclusion

The research is summarized and an answer to the research question is stated.

2. Geographic Data Sourcing

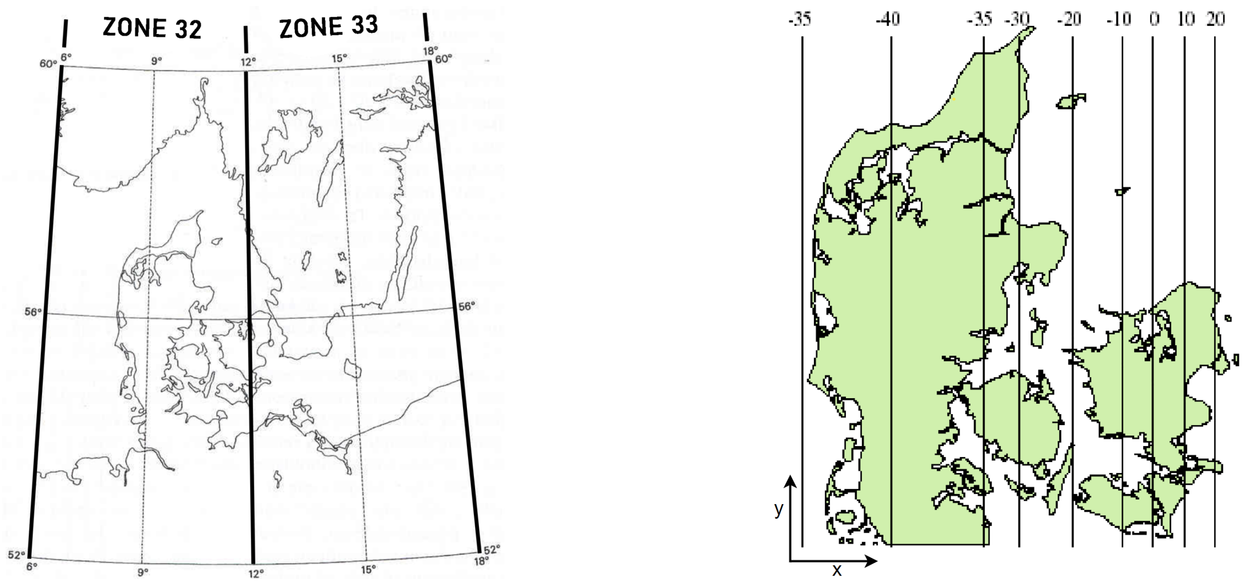
Geographic data is commonly used for city planning, navigation, and other private and commercial purposes. A wide selection of data is available to the public through APIs, either as open source or billed usage. The APIs supply data from geographic information systems (GIS), which can be configured in multiple ways, delivering data in different formats, quality, and versions. To achieve proper segmentation, it is important to understand what data is available and the specifications of such data.

2.1. Coordinate and Tile Systems

Data is typically requested using a geographic coordinate system for static APIs or a tile system for tile APIs, with APIs often limited to a subset of such systems. There exist a variety of coordinate and tile systems, covering either globally or regionally. The choice of system is especially important for raster data, such as maps and satellite images, as the spherical data is projected upon a square frame. This projection introduces a non-linear relationship between coordinates and real dimensions, of particular relevance for path planning, considering the dimensions of a robot.

The commonly used global coordinate system is EPSG:4326 (WGS84), covering the earth using latitude and longitude. The coordinate system is non-projected, representing positions on a spherical map. Such a system is optimal for GPS but requires a projection method for conversion to a 2D map. The industry standard for global services such as web-app map services, is using Web Mercator projection [6]. The projection is a spherical projection based on the radius at equator. This projection of the earth on a flat surface, causes region sizes to increase with distance to equator. The distortion becomes more pronounced at global view but decreases at an increasing zoom level.

In contrast, some APIs allow the use of regional coordinate systems, where EPSG:25832, also called Universal Transverse Mercator (UTM zone 32N), is frequently used in Denmark. This projected coordinate system is limited to a specific longitude interval, resulting in a narrower coverage area as seen in figure 2.1a. This allows for projection using a cylindrical transverse Mercator projection. The advantage of such is reduced distortion, increasing the accuracy of measurements. Despite the most eastern part of Denmark being within zone 33, government mapping services use zone 32 as a reference for the entire nation, increasing the interval of longitude. The UTM coordinate systems are frequently used for precise mapping, having coordinates expressed as x and y coordinates in the unit of meters. Due to the high latitude of Denmark, distortion is still a considerable factor, where the general guidelines of measurement error are seen in figure 2.1b. The available sources of documentation do not clarify if the error is equal along the x- and y-axis.



(a) UTM zone 32 & 33. [7].

(b) Projection error (cm per km) [7].

Figure 2.1: EPSG:25832 projection specification.

An example of two images requested using EPSG:25832 and EPSG:4326 are seen in section 2.2.1, figure 2.2.

A selection of map services uses tile systems to provide pre-processed tiles, reducing server load and processing time, in contrast to generating images from a bounding box. Tiles are maps divided into indexed sub-maps, with the quantity and tile size defined by a zoom level. A higher zoom level increases the quantity of tiles and reduces the area of such. Being pre-processed, tiles have a fixed maximum resolution, meaning a higher zoom level offers increasingly detailed views. Geodata is projected onto tiles, with the projection method being set by the individual map services.

2.2. Raster Map Services

Raster map services provide data in an image format, typically specified by the request parameters. Raster data for path planning can be categorized into orthophoto, navigational maps, and architectural maps.

2.2.1. Ortophoto Data

Three of the major map service providers for orthophoto data in Denmark are Google, Mapbox, and Dataforsyningen (Danish Data and Map Supply). Each provider offers distinct features for data and API capabilities.

Google and Mapbox offer globally covering map services supplying satellite images. Data is sourced from various providers [8] [9], some of which either source from or supply to Dataforsyningen. This means the three providers may use some of the same data. The web app version of Google Maps specifies the image source for a tile, revealing that most image data in central Aarhus is acquired from Maxar Technologies, an American space company.

Dataforsyningen's orthophoto coverage is limited to the kingdom of Denmark, including Denmark, Greenland, and the Faroe Islands. Dataforsyningen sources their Danish orthophoto data from GeoDanmark, a Danish organization facilitating data for Danish agencies.

Google and Mapbox provide a static API using EPSG:4326 and a tile API using the global tile system. All maps are projected using Web Mercator projection. Dataforsyningen offers tile and static APIs, using a national tile system and a wide selection of coordinate systems, including EPSG:4326 and EPSG:25832. Some coordinate systems are specified with a projection, though Dataforsyningen does not specify which projection method is used for the tile system and EPSG:4326.

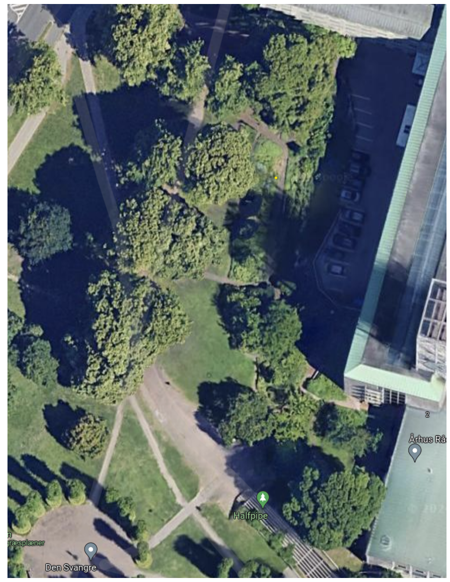
Google's static API takes a center point and a zoom level as input, lacking support for a bounding box. The API allows for a maximum output size of 640x640 pixels, requiring a high level of zoom to capture details. Dataforsyningen and Mapbox static APIs takes a bounding box and resolution as argument. Dataforsyningen's static API has a resolution limit of 10,000x10,000 pixels, and Mapbox's static API is limited to 1280x1280 pixels.

All tile APIs take a tile index defined as x and y and returns an image. Google returns a png image of 256x256 pixels. Mapbox can return multiple formats including png and jpg, and offers resolutions of 256x256 and 512x512 pixels. Dataforsyningen offers png or jpg images with a resolution of 256x256.

Orthophoto data must represent the current reality as modern cities continuously change their layout. Google's web app displays the year of obtaining their data. It is observed that most images are updated in the current year of 2024, though inconsistencies are observed between the images and reality. Current roadwork that has been ongoing since 2023, at a specific location in Aarhus, is not shown, indicating the data may have been acquired from a source in 2024 but was not captured in 2024. Dataforsyningen's API allows users to specify which annual dataset to request from, with 2023 being the current default and the newest. This does not guarantee the data is from 2023, but Dataforsyningen specifies that no data is captured before 2012. Mapbox does not specify the age of their data.

Google's and Mapbox's images are generally captured in the summertime, in contrast to Dataforsyningen's images being captured in spring. An advantage of capturing images in spring is the lack of vegetation covering the layout, such as leaves of trees. A comparison of a park in Aarhus is seen in figure 2.2, highlighting the advantage of not having leaves, making paths more visible. It may also be noted that the images have different dimensions, though the bounding box is the same. This shows a significant difference between using Web Mercator projection and EPSG:25832.

(a) Spring satellite image [10].



(b) Summer satellite image [11].

Figure 2.2: Orthophoto image season comparison.

As seen in figure 2.2, orthophoto images contain shadows and perspective. Shadows change the color of areas cast upon. The perspective of buildings can cover sidewalks and other details. Weather does not affect the images, as these are captured when the sky is clear. The maps contain noise in the form of objects and people.

Additionally, Google offers orthophoto images at a 45-degree angle capturing details that would otherwise be blocked by structures, in contrast to regular orthophoto images captured closer to normal to the surface. This introduces issues regarding handling locations, as such data would be complex to relate to a coordinate system.

2.2.2. Navigational Maps

The two considered providers of navigational maps are OpenStreetMaps (OSM) and Google Maps. MapBox is not considered, as it sources all maps from OSM. OSM does not specify its sources of data for maps of Denmark [12]. OSM and Google Maps both provide maps of infrastructure, containing details such as roads, sidewalks, and buildings. The maps are mainly intended for navigational use, lacking accuracy in regards of scaling entities. By comparing the maps from both services to a satellite image as seen in figure 2.3, it can be observed that these are far from accurate, as the dimensions of roads and pavements do not match the reality.

(a) Satellite image [11]. (b) OpenStreetMaps. [13] (c) Google Maps.[11]

Figure 2.3: Comparison of navigational map sources in Aarhus, Denmark.

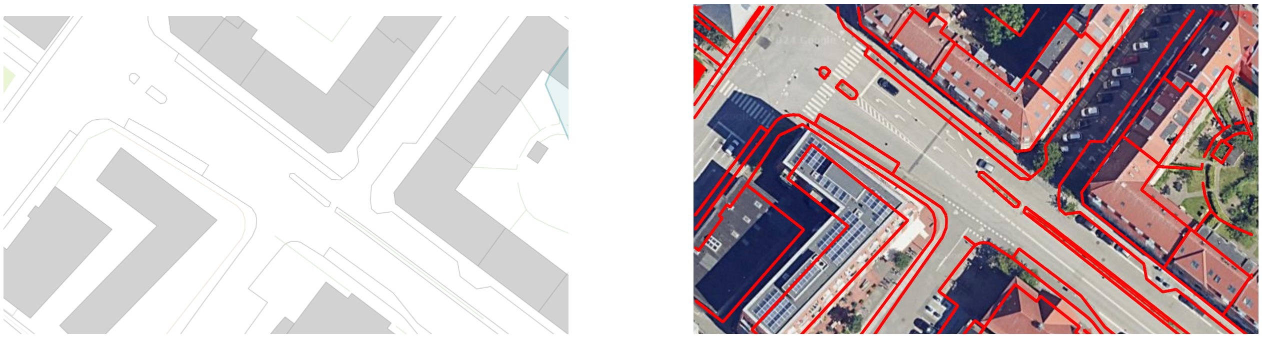
Google is working on optimizing its maps towards more accurate dimensions by implementing the "street-level details" feature [14]. This feature gives a more accurate scale of roads, sidewalks, crosswalks, and building boundaries. However, the feature is yet to be available in Aarhus and is currently only available in large cities like London and partially in Copenhagen. Google Maps and OSM use EPSG:4326 projected with Web Mercator [15]. Projection can be disabled, allowing for custom implementation of other projection methods.

2.2.3. Architectural Maps

Dataforsyningen offers a comprehensive range of APIs, including Forvaltningstjenesten, which contains nearly all architectural data available from GeoDanmark [16]. The map service offers technical data projected on top of a base map. The base map is an administrative map used for city planning, capturing details such as sidewalks, roads, buildings, waterways, and more. Henceforth, maps generated by Forvaltningstjenesten will be referred to as administrative maps.

Forvaltningstjenesten allows the generation of static maps up to 20,000 x 20,000 pixels and tiles of 256 x 256 pixels, both in jpg or png format. The static API supports various coordinate systems, including EPSG:4326 with a non-specified projection and EPSG:25832. The tile system has a non-specified projection.

An administrative map from the same area of Aarhus as section 2.2.2, figure 2.3 is seen in figure 2.4. The edges of the administrative map are laid over a satellite image, where the contours follow walkways and buildings significantly more accurately than OSM and Google Maps. It must be noted that the satellite image is affected by perspective, whereas the administrative map focuses on the base of the building. Overhangs from buildings can be included in the administrative maps.



(a) Administrative map. [16]

(b) Administrative map contours over a satellite image. [16] [11]

Figure 2.4: Administrative map and comparison to a satellite image.

The administrative map can be generated with centerlines for roads, labeled waterways, addresses, and much more.

The administrative map does not indicate whether a road or bicycle lane is private or public, which is information available from the Danish Road Directorate through a web app [17]. There is no public documentation of any API. By inspecting the requests made by their web app, it indicates there is an API following the same standard as for Forvaltningstjenesten, and it does not require a token. The returned map is a map of centerlines for roads color-labeled to indicate private or municipal ownership.

Property boundaries can also be accessed through a dedicated API from Dataforsyningen, called Matriklen [18]. It allows for accessing boundaries without an administrative map background and accessing more information about properties. Boundary maps can be generated as static maps up to 10,000 x 10,000 pixels in jpg or png. The service supports various coordinate systems, including EPSG:4326 with a non-specified projection and EPSG:25832.

2.3. Discussion and Conclusion

Map Services offer a wide selection of coordinate and tile systems. The choice of coordinate system and projection method has an impact on the accuracy of data, where EPSG:25832 is commonly used for city planning in Denmark, due to the higher precision. The dimensional error of EPSG:25832 is specified through an imprecise illustration without declaring if the error is equal along both axes. Some map services do not specify which projection is used.

Tile services have a low resolution, requiring a high zoom level. Static map services have a limited resolution not lower than 10,000 x 10,000 pixels, except Google Static API, allowing for a higher level of detail. Tile services are faster than static map services as these are pre-processed. This is not critical as the data is segmented before a robot is deployed in the field.

All Danish map services seem to follow the overall map service standard, though these offer limited or no documentation.

Orthophoto map services differ in coordinate systems, projection methods, data sources, and season of captured images. The spring season of Dataforsyningen has an advantage over Google and Mapbox's summer images, by having a reduced density of vegetation covering paths. All orthophoto map services have the disadvantage of containing perspective, shadows, and objects.

Google Maps and Open Street Maps' navigational maps have poor accuracy compared to administrative maps. Almost all GeoDanmark architectural data is available through Dataforsyningen, where Forvaltningstjenesten's map service contains most, only lacking road ownership and displaying data without administrative maps.

The final choice of data depends on the requirements of the segmentation models.

3. Orthophoto Segmentation

Path planning relies on accurate segmented maps derived from suitable data, ensuring precise coverage of the area by a robot. The selection of appropriate geospatial data, choice of segmentation models, testing, and evaluation are described in this chapter.

3.1. Related Work

Universal segmentation models are explored, where the Segment Anything Model (SAM) by Meta AI, is the state of the art. The model is proposed in the paper "Segment Anything" by Kirillov et al.[19]. SAM is trained on 11 million images with 1.1 billion masks, containing various scenarios. Orthophoto images are not listed in the dataset.

The model utilizes an encoder and decoder, to separate the image into segments. The model does support text prompts to segment and label objects. This feature is yet to be released [20], thus the returned data is unlabeled.

Segmentation of sidewalks is a relatively under-researched area compared to segmentation of roads. The paper "A Survey of Deep Learning Road Extraction Algorithms Using High-Resolution Remote Sensing Images", by Mo et. al [21], evaluates and compares available deep-learning models for road segmentation. The paper highlights the challenges of extracting roads due to various road types, and buildings and vegetation occluding roads.

The paper concludes that fully supervised learning achieves the highest accuracy but lacks generalization compared to semi-supervised learning. It's stated that supervised learning models use a large amount of labeled data for training, requiring manual labeling. Available pixel-labeled datasets are listed, though these are all limited to areas of the United States of America. It's stated that point- or line-labeled data, such as vector data is broadly available compared to raster labeled data. It's stated that complex models have higher accuracy and concluded that LDANet [22] is the most accurate model, demonstrating precision and recall of 97.55% and 97.07% on the Massachusetts roads dataset. It must be noted that the model is evaluated on data covering rural roads, not being subject to buildings occluding areas.

A model trained specifically for sidewalks is proposed in "Mapping the walk: A scalable computer vision approach for generating sidewalk network datasets from aerial imagery" by Hosseini et al. [23]. The software containing the model is called Tile2Net.

Tile2Net utilizes a semantic segmentation model trained to determine sidewalks, crosswalks, roads, and backgrounds. The model is based on the HRNet-W48 model, a deep high-resolution network consisting of multiple blocks for feature extraction and finally a convolutional network. The result of the convolutional network is fed into a connected component mapping algorithm to detect polygons. Centerlines are inferred from polygons to create a graph.

The pipeline of the program is seen in figure 3.1, where raster data is segmented and converted to polygons, simplifying the raster areas and removing noise. The polygons are processed into graphs, which are not useful for complete coverage path planning, thus the polygons will be considered the final result.

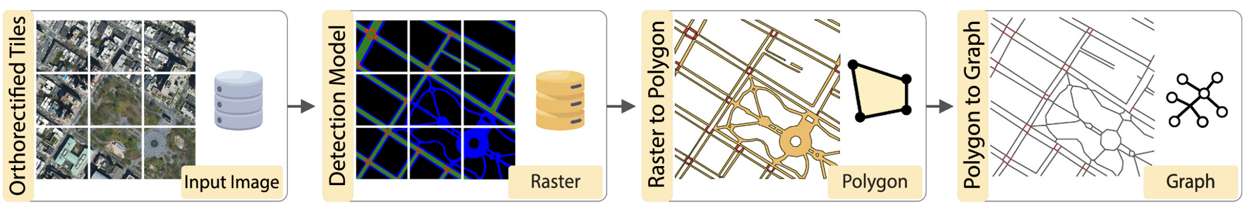


Figure 3.1: Tile2Net pipeline [23].

Tile2Net supports tiles of 512×512, 1024×1024, and 2048×2048 pixels. Tiles with 1024x1024 achieve the best results by having context to distinguish between classes. The software has a stitching function to combine low-resolution tiles into high-resolution tiles.

The model is trained on labeled data from orthophoto images of Massachusetts, Washington DC, and New York. The segmentation performance metrics are seen in table 3.1.

Table 3.1: Performance metrics for Tile2Net.

Label	**IoU (%)**	**Precision**	**Recall**
Sidewalk	82.67	0.90	0.91
Road	86.04	0.86	0.86
Crosswalk	75.42	0.97	0.96
Background	93.94	0.92	0.94

Though the performance metrics are high, it must be noted that the model is trained and tested in the United States of America, where infrastructure differs from Denmark, often having concrete sidewalks that have brighter colors. The paper emphasizes this concern, stating that the model can be used for transfer-learning on a dataset for another region.

Tile2Net is a complete system with implemented API's to source orthophoto data, limited to five American states. The program does not offer the use of orthophoto images of Denmark. Tile2Net is limited to the use of EPSG:4326 with Web Mercator projection.

Another approach to segmenting sidewalks is proposed in "Segmentation of Occluded Sidewalks in Satellite Images" by Senlet et al. [24]. The segmentation algorithm by Senlet et al. is unnamed and will hereafter be referred to as OSSA, Occluded Sidewalk Segmentation Algorithm. The purpose of OSSA is to overcome the challenges of sidewalks occluded by trees, shadows, and buildings. The approach uses orthophoto data and road orientation data.

Trees are classified using a random forest classifier, trained on labeled data. Tree shadows are classified using saturation-based color segmentation. The orientation of tree shadows is estimated by checking all possible angles and determining the angle where an extended tree region covers the shadow the most.

Sidewalks are segmented on SV color space (Saturation and values) but contain false positives. These are filtered using orientation filters to classify segmented pixels as sidewalks, buildings, or non-linear structures. Non-linear structures and buildings are removed from the segmentation. Sidewalk regions are grown by an appearance-based approach, determining connected pixels with similar appearance in SV color space. Regions that differ significantly from others are removed.

Sidewalk probabilities are determined by an undefined algorithm with reference to a non-public paper. Sidewalk branches are continued in occluded areas using inpainting algorithms, growing sidewalks in areas with respect to four principles: parallel to roads, moving towards the endpoints of other sidewalks, prior sidewalk probability, and continuing straight.

The framework of the algorithm is seen in figure 3.2. The algorithm also determines crosswalks.

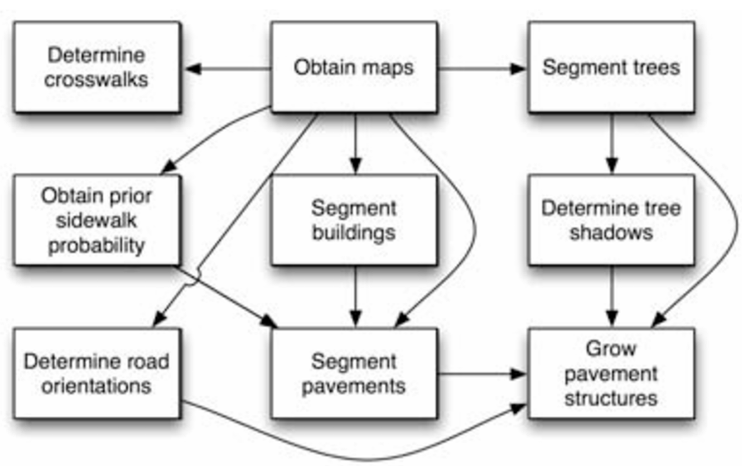


Figure 3.2: OSSA framework overview [24].

Examples of structure growth are seen in figure 3.3. The segmentation outputs are white and structure growth results are red.

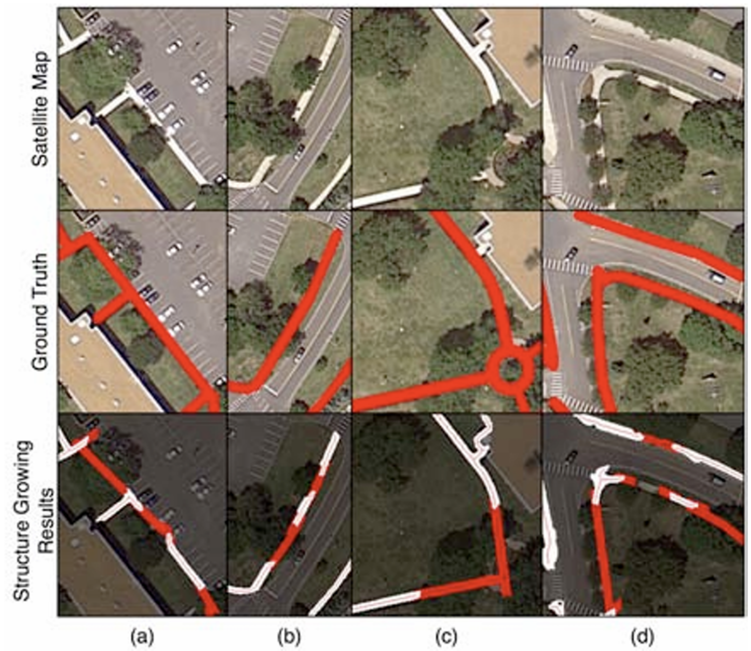


Figure 3.3: Structure growing example. (a) Continuation of linear structures on small edges. (b) Straight continuation (c) Joining of concealed intersections. (d) Growing parallel to roads. [24].

The performance metrics are determined by evaluating the algorithm on images from Rutgers Busch Campus in New Jersey.

Table 3.2: Performance metrics for OSSA

True Positive	False Negative	False Positive	Precision	Recall
41/54	13/54	6	87%	76%

The paper concludes that the initial sidewalk detection can be optimized to achieve higher accuracy.

3.2. Choice of Segmentation Models

The documentation of the universal segmentation model SAM doesn't state that satellite images are included in training and it does not have any features to segment occluded areas. SAM is publicly available and the output may be post-processed using functions of OSSA to segment occluded areas. The segments of SAM are not labeled as the prompt feature is yet to be released. Labels may be determined from color segmentation or inferred from shapes. SAM is chosen for testing, but post-processing is not implemented as SAM does not segment properly, as described in section 3.3.

The road segmentation model based on LDANet has a high precision on roads. The model would require transfer learning to segment sidewalks, which is not possible as labeled sidewalk data of Denmark is not available. A dataset for training would require a significant amount of samples to avoid overfitting, but labeling of data in that scale is not in the scope of this thesis. The model could be trained on the American datasets used for Tile2Net. However, LDANet is not publicly available.

Tile2Net outperforms OSSA in precision and recall, though it must be noted that these models are tested on separate images. Tile2Net is limited to use with a selection of APIs that don't cover Denmark. The source code is available from GitHub and may be modified to enable the use of other APIs. OSSA is not publicly available and would require development based on the paper. OSSA handles occluded sidewalks, which is a feature that could be implemented in Tile2Net.

The research focuses on segmenting public sidewalks, excluding private sidewalks, which is a feature that could be implemented in post-processing for SAM and Tile2Net, as these are not trained for such filtering. This is not implemented due to the low performance of the models as described in section 3.3 and 3.4.2.

The two selected segmentation models for testing are SAM and Tile2Net.

3.3. Segment Anything Model

Segment Anything Model is available as a Web App and Python module, but also as a geospatial-optimized version through the SAMGeo module [25]. SAMGeo uses tiles and segments these in batches without stitching. The model is tested using the default satellite source, which isn't specified in the documentation. The result is seen in figure 3.4, showing very limited ability to segment sidewalks. Only a single stretch of sidewalk is detected and the rest are either ignored or combined with roads or buildings.

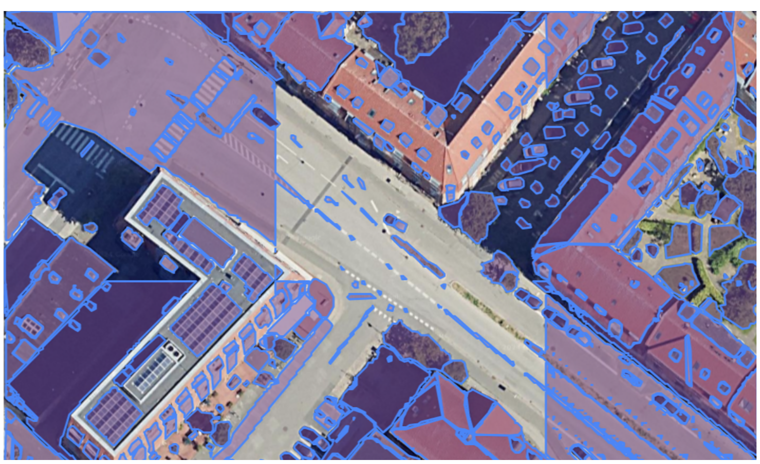


Figure 3.4: SAMGeo segmentation result.

Due to the poor segmentation of sidewalks, the model is not further implemented and optimized.

3.4. Tile2Net

Tile2Net works exclusively with tiles from a selection of APIs. To enable a free choice of Tile APIs, a custom data preparation function is developed. The function fetches tiles from a chosen tile source and stores them in the folder structure as specified by Tile2Net documentation [26]. Tile2Net stitches the tiles to a higher resolution and performs segmentation.

The custom data preparation function is tested on Google data from the Boston region to validate its functionality in comparison to the default APIs of Tile2Net. A comparison of the two resulting maps is seen in figure 3.5. The segmented polygons are laid on top of a Google satellite image for reference. The results verify that the custom API integration does work and segmentations are performed as for the default Tile2Net APIs.

(a) Built-in default API segmentation.

(b) Google API segmentation.

Figure 3.5: Compatibility testing of API integration. Blue = roads, sidewalks, and crosswalks.

As seen in figure 3.5, Google orthophoto data fails to segment sidewalks occluded by trees. It would be preferable to use data from spring with minimal vegetation. The spring orthophoto API from Dataforsyningen uses a regional tile system, which is attempted to adapt to the global tile system. The tile indices are converted to an estimate of the global index, but the nearest offset is not an integer, so the two systems can't be aligned. Therefore, Dataforsyningen is not used. Conversion of regional tile systems would require stitching and splitting of tiles and conversion to Web Mercator projection.

Mapbox tiles are segmented, using the max resolution of 512x512 pixels, as seen in figure 3.6a. Google tiles are segmented, using the max resolution of 256x256 pixels, as seen in figure 3.6b. Both tile services are used with a zoom of 20, as this is found to give the best results through iterative testing and is also recommended by the developers of Tile2Net [26].

(a) Mapbox segmentation result.

(b) Google segmentation result

Figure 3.6: Compatibility testing of API integration. Red = sidewalks

The resulting segmentations are evaluated on a manually drawn ground truth map. Google performs with a precision of 0.22, recall of 0.38, and accuracy of 0.89. Mapbox performs with a precision of 0.22, recall of 0.36, and accuracy of 0.9. The performance metrics are nearly equal and Google is chosen as the used data source due to the slightly higher recall.

Tile2Net fails to stitch tiles in the region of Denmark, meaning the model will have less context for segmentation. Stitching is a part of the generate function, which is highly unstable and throws non-descriptive errors. The error has not been fixed, but stitching does work on Boston locations using data fetched through Tile2Net and the custom data preparation function.

3.4.1. Results

The Tile2Net is tested on ten distinct locations of Aarhus, each containing its distinct environments. Ground truth segmentation maps are manually drawn based on Google orthophoto images, maps from Dataforsyningen, physical inspection, and Google Streetview. Performance metrics are seen in table 3.3 and segmentation maps compared to ground truth are present in Appendix 1.

Table 3.3: Performance metrics for ten locations - Tile2Net

Location	Precision	Recall	Accuracy
1	22%	38%	89%
2	32%	10%	83%
3	29%	40%	89%
4	37%	41%	96%
5	49%	40%	91%
6	33%	51%	92%
7	29%	37%	93%
8	36%	27%	90%
9	7%	28%	95%
10	20%	58%	90%
Average	29%	37%	91%
Standard deviation	0.11	0.13	0.03

The segmentation results of Appendix 1 show a tendency of the model to poorly recognize sidewalks unless in an open area with distinct sidewalks. The model generates many smaller non-linear segments scattered around the map and straight edges on segments indicating that a tile border may have influenced the resulting segment. The model can't distinguish between public and private sidewalks and also tend to classify bicycle lanes as sidewalks.

3.4.2. Discussion and Conclusion

Tile2Net is able to use other than the default APIs, using custom functions. Though spring orthophotos may optimize the segmentation due to less occlusion of walking paths, such data for Denmark isn't supported without further processing to convert the data to the supported tile and projection system. Google is the best source of supported data, though MapBox data performs near to equal. The stitching feature of Tile2Net is highly unstable and not useable in Danish regions, resulting in segments being cut off due to tiles being processed separately. Tile2Net may benefit from post-processing using functions of OSSA to grow occluded sidewalks and filtering of public and private areas. Post-processing is not implemented due to the low-performance metrics and small scattered false positive areas that are likely to affect growing. Tile2Net shows an ability to segment with a precision of 29%, recall of 37%, and accuracy of 91%. The precision and recall have a large deviation.

3.5. Chapter Conclusion

Tile2Net and SAM are the two only publicly available models, and Tile2Net performs the best compared to OSSA. SAM is evaluated to see if it gives a good initial segmentation result with potential for use with post-processing. SAM is discarded due to poor performance.

Tile2Net is evaluated on ten locations of Aarhus but shows poor ability to segment sidewalks. It isn't possible to apply transfer learning to adapt it do Danish infrastructure, due to the lack of data. Post-processing could be applied to optimize the model, but due to scattered false positive segments complicating growth, this isn't further implemented.

4. Architectural Map Segmentation

In contrast to orthophoto data, being prone to occlusion and shadows, architectural maps are drawn illustrations representing details of the physical landscape. The following chapter proposes and evaluates a segmentation model to identify sidewalks based on this data.

The researched segmentation models presented in section 3.1 might be useful for the segmentation of architectural maps. However, as there is no labeled data for training, it is decided to investigate the use of image processing algorithms to combine the data of maps and segmentation. The developed segmentation model is referred to as AMS, Architectural Maps Segmentation. SAM, (Segment Anything Model) is considered as the maps are simpler than orthophoto, but poor segmentation performance is observed.

4.1. Theory

The segmentation model utilizes image processing functions, applied through the OpenCV library. The used methods are elaborated hereafter.

Noise filtration

Dilation is a morphological method that grows objects in a binary image by adding true pixels to the boundaries. Dilation uses a kernel to define the area to grow.

Erosion is the opposite of dilation. The method shrinks areas by setting pixels to false in an area determined by a kernel. Can be used to remove noise.

Opening morphology is the method of erosion followed by dilation. Erosion reduces the size of objects, effectively removing noise on a background. Dilation grows the remaining objects back to near original dimensions.

Closing morphology is dilation followed by erosion. Dilation fills holes in objects and erosion reduces the outer dimensions back to near original size.

A bilateral filter is used to reduce noise while preserving sharp edges. It considers both the pixels within range and pixel value similarity.

Median blurring is replacing the pixel value with the median of the neighboring pixels determined by a kernel.

Contour detection

OpenCV includes a findContours function, to determine the contours of an image based on a line following algorithm [27]. The function outputs contours in a hierarchy, with each contour represented by a set of points.

Thresholding

OpenCV offers thresholding to filter an image by pixel value ranges on both BRG, HSV, and greyscale. The library also includes OTSU's method, which automatically determines a threshold to separate the background and objects of a greyscale image.

4.2. Map Sources

Maps are sourced from Dataforsyningen and the Danish Road Directorate through various APIs. Common for all map services is the support of EPSG:25832, offering the highest precision in Denmark and most suitable for path planning. Tiles are offered by the used map services, but static maps are preferred as the services offer high resolution, and segmenting an entire map gives more context.

All map services require a resolution set in the API request. The aspect ratio can be calculated by the aspect ratio of EPSG:25832 coordinates, as these are in the unit of meters. The lowest resolution limit by the used map services is 10,000. All data is requested as png, to be supported by OpenCV.

A base map and a structure map are requested from the Forvaltningstjenesten API offered by Dataforsyningen. The base map contains administrative details of sidewalks, buildings, and other details. The structure map is another version of a base-map with buildings and other structures colored orange, which is easier to segment. As both are base maps, these contain details that are irrelevant to the segmentation and must be filtered off. Forvaltningstjenesten does not offer many settings regarding base maps.

A property boundary map is requested from Dataforsyngen Matrikel API, set up to only include lines of properties and no other data. This API is chosen over Forvaltningstjenesten offering the same data, because the map contains no other details.

Routes are requested from the Danish Road Directorate, representing bike paths and roads, and the ownership of these. The layer "CVF:veje" is requested. There is no documentation of the API, but it is static and supports a resolution of 10,000 x 10,000 pixels.

Orthophoto images are fetched from Dataforsyningen Orto-Forår API. This is the only orthophoto service offering EPSG:25832, which is important to maintain the same projection amongst maps.

4.3. Segmentation Model

The segmentation model processes a base map through a sequence of filter functions. Each filter function takes maps as input to segment unwanted areas and excludes these from the base map. The filtered base map is converted to binary and filtered for noise. The workflow is illustrated in figure 4.1.

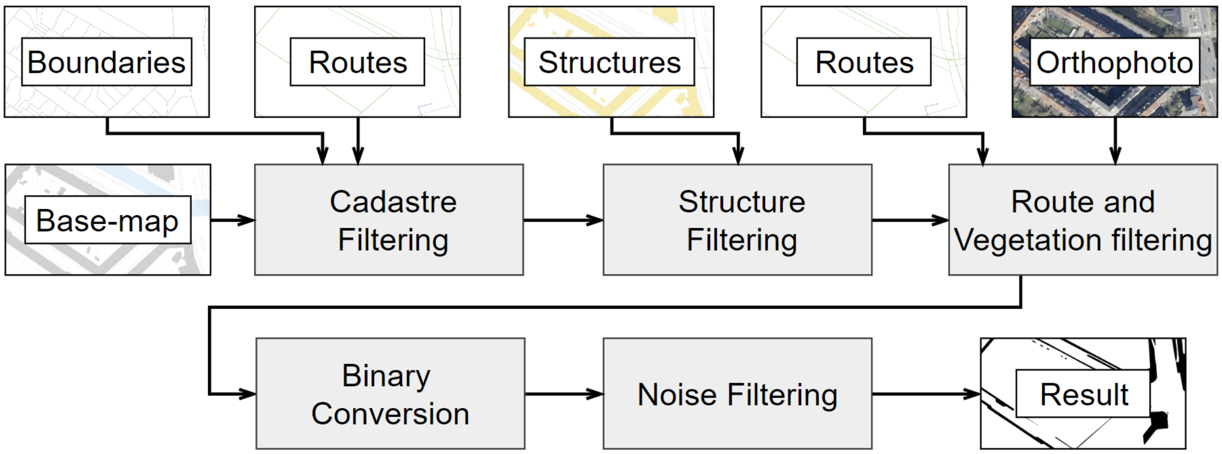
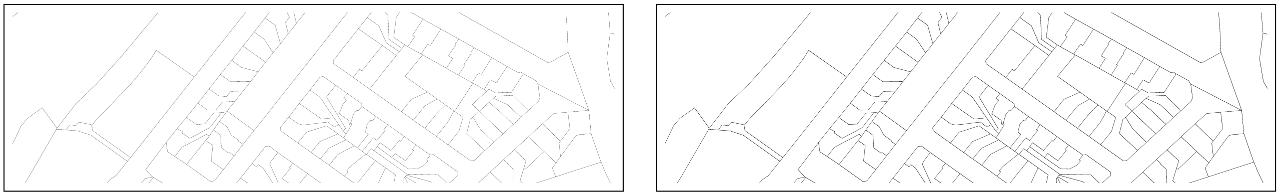


Figure 4.1: Segmentation Sequence.

Each filter function consists of segmentation and filtering functions, which are detailed in the following sections.

4.3.1. Cadastre Filtering

Cadastre filtering removes all areas from the base map being private property, as these don't contain public sidewalks. The boundaries map is converted to greyscale as seen in figure 4.2a, to allow for segmentation of cadastres using the boundary lines. As a consequence of the low resolution and the map service's methods for resizing maps, these lines may not be continuous. The lines are enhanced and connected using dilation. The image is inverted before dilation, as dilation grows white pixels. Dilation is performed using a 2x2 rectangular kernel, effectively bridging gaps in lines. The choice of using a rectangular structural element for the kernel does not have an effect due to the low resolution of a 2x2 kernel. Finally, the binary map is inverted to the result seen in figure 4.2b.

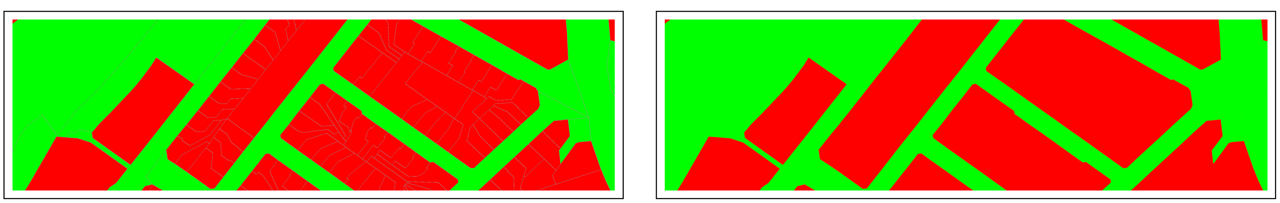


(a) Greyscaled boundaries. (b) Enhanced boundaries.

Figure 4.2: Boundary enhancement.

Contours are detected using the OpenCV "findContours" function, performing line following to create a contour representation of the map. This representation includes properties and public areas, each enclosed by contours. The areas of contours are used to isolate areas of the route map, giving a separate mask for each contour. The masks are iteratively checked for green pixels, indicating public areas containing public roads or bicycle lanes. If a cadastre contains green pixels, it is a public area; otherwise, it is a private area. Private areas are marked in red and public areas are marked in green, in a resulting mask as seen in figure 4.3a.

The contour algorithm treats lines as areas, resulting in lines having their own contours, leaving green areas in the result. These are filtered by size, checking if the contour area size is over 100 pixels, and deleted if too small. As lines are left uncolored, these remain black on the red and green map. Due to the thin width of lines, these are removed using smoothening. The lines are smoothened out using median blur with a kernel of 7, blurring each pixel based on the median value of pixels in a 7-pixel radius.



(a) Segmented properties. (b) Blurred segments.

Figure 4.3: Boundary segmentation.

The red segments are transferred from the mask to the base map using color thresholds, indicating areas not subject to defrosting, as seen in figure 4.4.

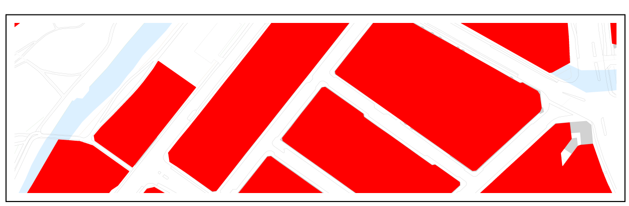
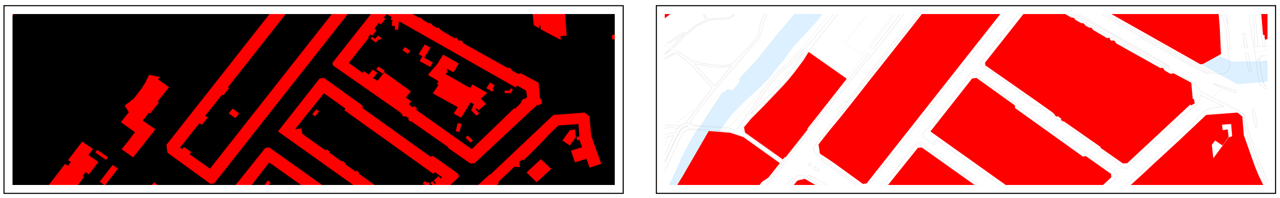


Figure 4.4: Cadastre filtered base map.

4.3.2. Structure Filtering

Most structures, often being buildings, are located within the private properties marked in section 4.3.1. Some structures are located on public property or a structure may extend beyond the private property. The structure map has structures colored orange, which are isolated in a mask using HSV color thresholding within the range of H=10-60, S=50-255, V=50-255. The resulting mask is seen in figure 4.5a and transferred to the base map in figure 4.5b.



(a) Segmented structures. (b) Filtered structures.

Figure 4.5: Structure filtering.

Some structures are buildings having an extended part reaching out over the sidewalk. These should not be included; however, can't be separated from the rest of the structure. The model is tested with and without filtering of structures in section 4.4.

4.3.3. Route and Vegetation Filtering

The base map contains waterways and green and brown lines, which are filtered off before contour detection as they aren't used and will add noise to the segmentation. Filtering is done using BGR (Blue, Green, Red) thresholds seen in table 4.1, replacing all positive matches with white pixels.

Table 4.1: BGR threshold values.

Color	**Lower**	**Upper**
Brown	200, 215, 230	210, 230, 240
Green	205, 230, 210	220, 255, 235
Blue	250, 235, 215	255, 245, 225

Some noise is left from filtering due to the low resolution and downsizing methods from the web service. Noise may be reduced if HSV filtering is used instead of BGR filtering. The noise is reduced using a bilateral filter, which has the advantage of preserving edges. The filter has a diameter of 5 pixels and a sigma color parameter of 255 to maximize smoothening.

The map is converted to greyscale and converted to binary with a threshold of 245-255 to filter edges. Edges are enhanced using dilation with a 2x2 kernel and inverted before and after as dilation works on white pixels. The result is seen in figure 4.6.

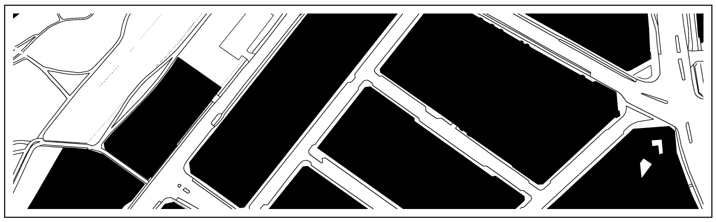


Figure 4.6: Filtered binary base map.

Contours are determined, to identify areas to check for routes and vegetation, using OpenCV "findContours" function. Contours are filtered with the requirement of being larger than 100 pixels or below half of the total image size, to reduce noise and contours covering the entire image. Each valid contour is iterated over, checking for routes and vegetation in the following two functions.

Road and Bike-path Filtering

The route map is converted to greyscale and converted to inverted binary with a threshold of 245-255, giving an image with true (white) indicating all routes.

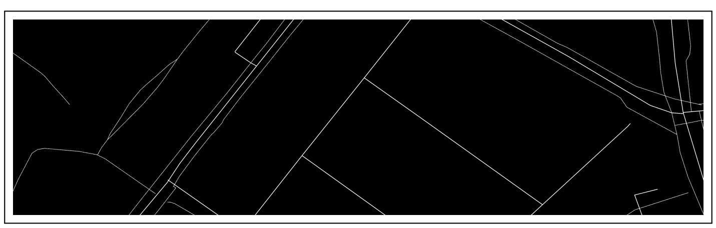
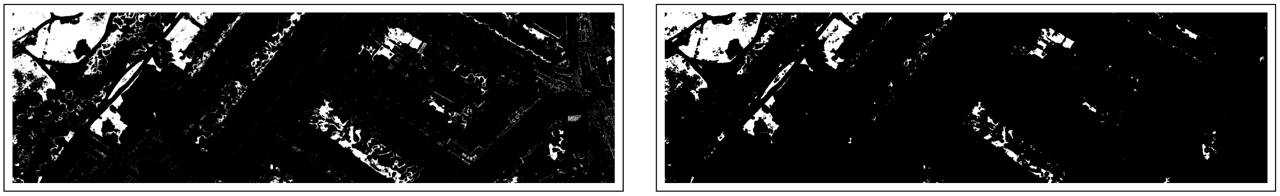


Figure 4.7: Binary route map.

Each contour of the base map is checked if it contains more than 0.1% true (white) pixels, indicating the area is a route and colored red on the base map. The threshold of 0.1% is found to be fitting, to allow routes crossing sidewalks.

Vegetation Filtering

The base map contains lines representing edges between vegetation and pavement; however, these do not contain enough information to classify vegetated areas, such as lawns. Satellite images are converted to HSV color scale and converted to binary using a threshold of H=42-72, S=5-250, V=5-250. The result of filtering is a mask indicating green colors with noise as seen in figure 4.8a. The mask is filtered using erode with a kernel of 5x5, resulting in the mask of figure 4.8b.



(a) Green vegetation mask. (b) Reduced green vegetation mask.

Figure 4.8: Vegetation mask.

The same method is applied to create a mask of waterways; however, the water is non-distinguishable from shade-covered roads, thus not implemented. The waterways illustrated in the base map are not used, as they would filter off any sidewalks on bridges crossing waterways.

Each contour is evaluated on having more than 10% green pixels, indicating it is a vegetated area, yet not falsely segmenting areas occluded with leaves. Vegetated areas are colored red, with the result seen in figure 4.9, also containing route filtering.

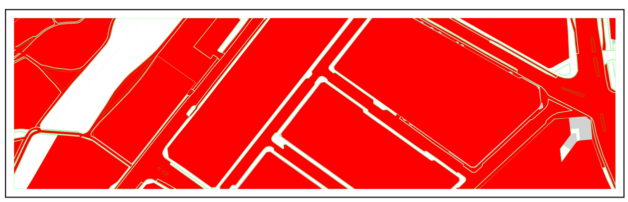
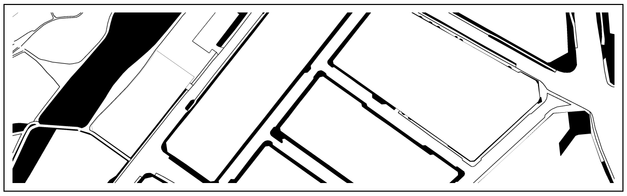


Figure 4.9: Base map filtered for routes and vegetation.

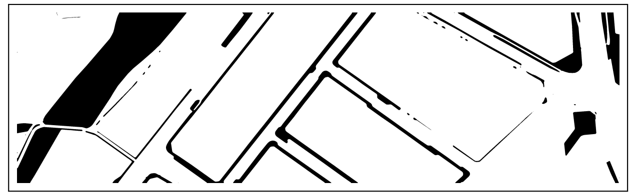
4.3.4. Binary Conversion and Noise Filtering

The base map is converted to a binary map using Otsu's Binarization to convert red areas to black and other areas to white. Black and white indicating defrosting and non-defrosting areas. The binary map contains noise from the previous functions. A series of noise filtering methods are performed, to reduce noise while keeping the dimensions of defrosting areas.

An open morphology transformation is executed, which is erosion followed by dilation. This reduces noisy black pixels in white areas. Followed by a closed morphology transformation, being dilation followed by erosion, which reduces white noise in black areas. Both operations use a 4x4 kernel of ones. Finally, a median blur with a kernel of 15 is used to smoothen out areas of insignificant size. The kernel size is determined by iterative inspection of the resulting map. Figure 4.10 shows the map before and after noise filtration.



(a) Base map before noise filtration.



(b) Base map after noise filtration.

Figure 4.10: Noise filtration comparison.

4.4. Results

The segmentation model is evaluated on the same locations as Tile2Net. As the model is working with EPSG:25832, the ground truth maps of section 3.4.1 can't be used. The ground truth maps are redrawn on the same principles, yet may differ in accuracy due to being manually drawn.

The model is tested with and without structure filtering to determine if the filter contributes positively. Results are seen in table 4.2 and 4.3.

Table 4.2: Performance metrics for AMS with structure filtering.

Location	Precision	Recall	Accuracy
1	25%	62%	88%
2	27%	5%	85%
3	68%	63%	95%
4	38%	89%	96%
5	70%	94%	96%
6	27%	85%	89%
7	12%	79%	74%
8	35%	19%	91%
9	6%	96%	83%
10	44%	86%	96%
Average	35%	68%	89%
Standard deviation	0.20	0.30	0.07

Table 4.3: Performance metrics for AMS without structure filtering.

Location	Precision	Recall	Accuracy
1	24%	65%	88%
2	21%	5%	84%
3	67%	63%	95%
4	38%	89%	96%
5	67%	96%	96%
6	27%	85%	89%
7	12%	80%	73%
8	11%	22%	81%
9	6%	96%	83%
10	45%	88%	96%
Average	32%	69%	88%
Standard deviation	0.21	0.30	0.07

The performance metrics show that AMS using structure filtering has slightly higher precision and accuracy than AMS without structure filtering. It is determined by inspection and comparison of segmented maps, that the main contributor to increasing precision and accuracy is filtering off buildings not part of a private property. The use of structure filtering reduces recall slightly.

The model lacks filtering of waterways and railways, both being labeled as sidewalks.

Sidewalks and bicycle paths are observed to be incorrectly segmented due to incorrect mapping of routes, located on a sidewalk or outside a bicycle path.

Some edges detected as areas from contour detection, are still present and segmented as sidewalks.

Detailed segmentation results are presented in appendix 2.

4.5. Discussion and Conclusion

The model is evaluated with and without structure filtering. Structure filtering increases precision and accuracy while recall is slightly decreased. The decrease in recall may be due to overhangs from buildings covering sidewalks, thus filtering off areas that should be defrosted. It is observed that some structures are buildings outside of private properties, which are segmented as sidewalks without the use of structure filtering. This may explain why structure filtering increases precision and accuracy, by reducing false positive sidewalk segmentations.

Binary maps are used as false (black) being sidewalks and true (white) being other areas. This implies inverting binary maps for proper use of filtering methods, which could be avoided if the notation was changed to false (black) being other areas and true (white) being sidewalks. It may optimize the model regarding computing time.

Waterways and railroads are being labeled as sidewalks, due to a lack of data to filter these categories. The base map does show waterways but overlaps with bridges. Using this data may cause sidewalks on bridges to be labeled as non-sidewalks.

The result contains noise from the contour detection of edges. The contour detection algorithm is not optimal for this usage as it detects edges as areas. Another approach with an algorithm for vectorizing edges may be more appropriate.

The route map causes some sidewalks to be ignored and some other areas to be labeled as sidewalks. This is observed to be due to the routes not following the real path.

The model is evaluated on ground truth maps drawn from the interpretation of multiple maps and physical observations. Some of the maps used for this are used to construct AMS, which may introduce a bias in the ground truth and affect the reality of the performance metrics.

5. Segmentation Model Comparison

Tile2Net and AMS with structure filtering are compared on performance metrics in table 5.1.

AMS shows a 6% higher precision but with a higher standard deviation. The segmentation results in appendix 2 show that the main areas segmented as false positives by AMS are waterways, railways, vegetated areas, and properties that are not within a private boundary. These are only present in some maps with low precision, causing a high standard deviation. Tile2Net has false positive areas scattered around all maps, giving a lower precision and lower standard deviation. Tile2Net cannot distinguish between private and public paths, resulting in the segmentation of private paths, being false positive.

AMS shows a 31% higher recall by recognizing sidewalks of most maps significantly better than Tile2Net. AMS does miss most sidewalks in a few maps, due to issues discussed in section 4.5, contributing to a standard deviation of nearly 2.5 times that of Tile2Net.

Tile2Net has a 2% higher accuracy compared to AMS and with a lower standard deviation. The lack of ability for AMS to properly segment waterways, railways, vegetated areas, and properties may cause this and as the issue is present as a large area in a few maps, it may explain the high deviation.

Table 5.1: Performance metrics comparison for Tile2Net and AMS.

Metric	**Tile2Net**	**AMS**
Precision		
Average	29%	35%
Standard deviation	0.11	0.20
Recall		
Average	37%	68%
Standard deviation	0.13	0.30
Accuracy		
Average	91%	89%
Standard deviation	0.03	0.07

By inspection of the segmentation results of appendix 1 and 2, it can be observed that Tile2Net segments many unconnected areas, where the segmented areas of AMS are generally more connected as seen in the example of figure 5.1.

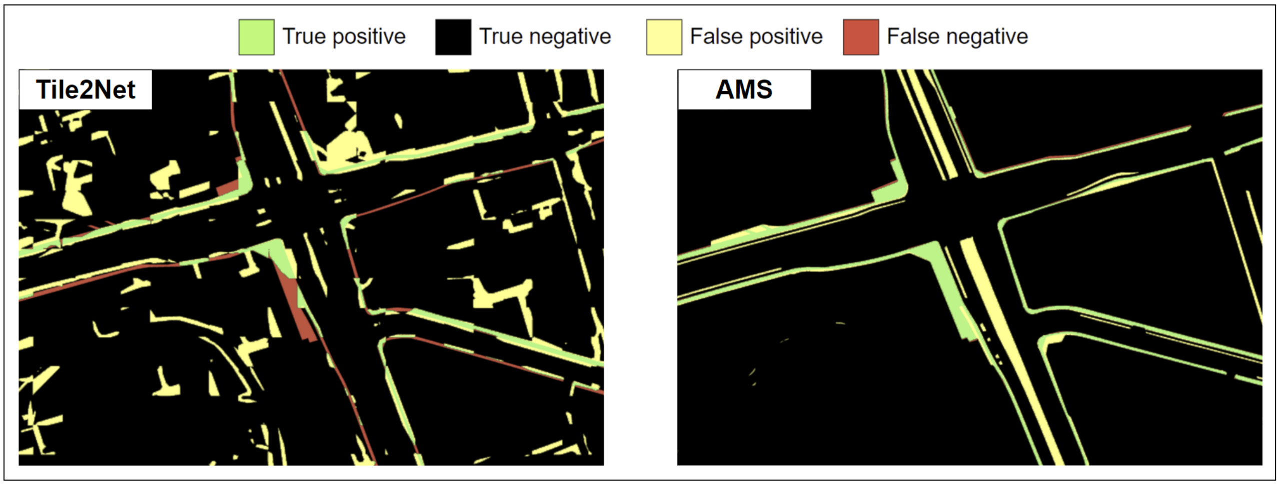


Figure 5.1: Comparison of Tile2Net and AMS Segmentation.

5.1. Discussion and Conclusion

Tile2net is a more generally stable model over AMS, due to the lower standard deviations. AMS does perform better on average, especially regarding recall. Segmented areas are more connected than Tile2Net, being an advantage regarding path planning.

Though Tile2Net has a lower standard deviation, the average performance of AMS is significantly higher, thus AMS is a better model for segmentation of sidewalks.

6. Map Suitability for Path Planning

Path planning of the resulting binary raster maps from segmentation is tested to ensure the format and notation of maps are suitable.

A proper path planning algorithm for defrosting isn't publicly available. A fitting path planning algorithm is expected to be complex and factor in robot kinematics, refilling and charging stations, road crossings, and the ability to move in all directions.

The maps' suitability is tested on a simpler coverage path planner developed by Renato Fernandes Rodrigues [28], referred to as CPP. The algorithm is chosen on being publicly available, python based, and independent of ROS.

The CPP uses a mix of search algorithms, including A* with a selection of heuristics, to find the complete coverage path with minimal cost in the form of distance. The action space includes up/down/right/left movement separately.

CPP works on maps in the form of nodes represented as numbers in a NumPy array. The representation is seen in figure 6.1.

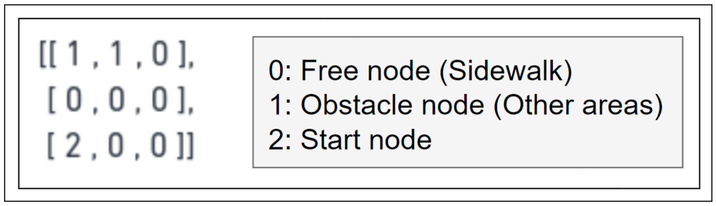


Figure 6.1: Node representation in NumPy Array.

6.1. Map Conversion

The segmented maps must be converted to a NumPy array. As each element of the array is a node covered by the robot, the width of a node must equal the width of the robot. The width of the bounding box is determined by EPSG:25832 coordinates in the unit of meters.

$$\text{areaWidth} = \text{max\_X} - \text{min\_X}$$

The downsized decimal width is determined, with a robot width assumed to be 0.5 meters.

$$\text{newMapWidth} = \frac{\text{areaWidth}}{\text{robotWidth}}$$

A scaling factor is calculated on the relation of the original map width and the downsized map width.

$$\text{scaleFactor} = \text{int}\left(\frac{\text{originalMapWidth}}{\text{newMapWidth}}\right)$$

The downsized map width and height are determined. Floor division is used, as pixel dimensions must be integers. An extra pixel is added to account for situations of an uneven scale factor.

$$\text{newMapWidth} = \left\lfloor \frac{\text{originalMapWidth}}{\text{scaleFactor}} \right\rfloor + 1$$

$$\text{newMapHeight} = \left\lfloor \frac{\text{originalMapHeight}}{\text{scaleFactor}} \right\rfloor + 1$$

Downsizing is done by combining a group of pixels in the width and height of the scale factor. These are combined by checking if the group contains black pixels representing sidewalk. If the group contains black pixels, the combined pixel will be black, otherwise it will be white. The method is illustrated in figure 6.2. Downsizing by this method ensures that areas of sidewalk are not lost, but precision decreases as white pixels are represented as black. Having a lower precision while maintaining recall is preferred, as the robot is assumed to be capable of performing collision avoidance.

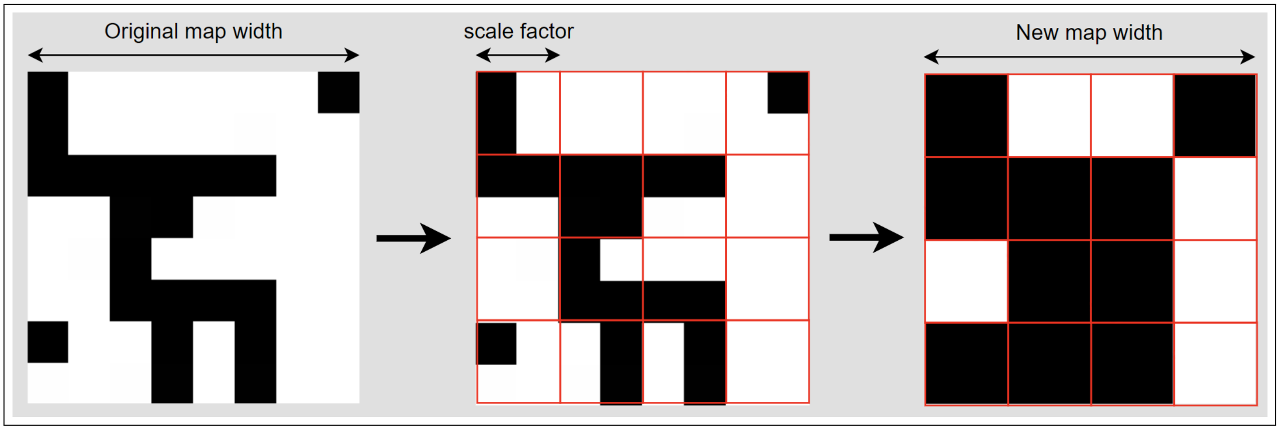


Figure 6.2: Downsizing of segmented map.

The downsized map is converted to a NumPy array by iterating over each pixel. Black pixels are written as value 0, white are written as value 1, and the first black pixel is written as value 2, being the start location for test purposes. The start position would be more optimal at a charging station.

6.2. Testing and Results

Testing is performed on the ground truth maps to avoid noise. CPP can only handle connected areas, thus crosswalks are drawn onto the ground truth maps.

Path planning is performed on Aarhus location ten. Due to the high quantity of nodes, the paths of figure 6.3 are hard to interpret, yet it is clear that no nodes are white, meaning all nodes are covered.

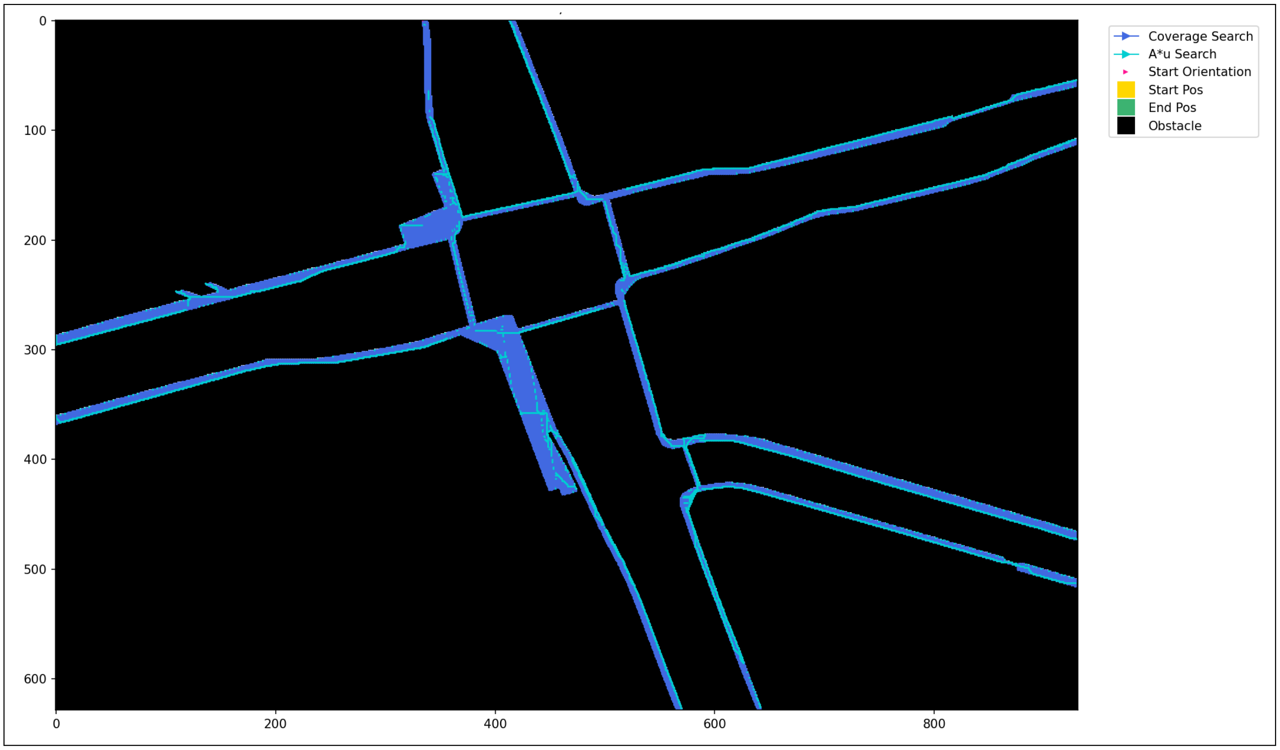


Figure 6.3: Aarhus location ten path planning.

Path planning is performed with a high scaling factor to analyze the workings of the algorithm, showing the paths in figure 6.4. The paths show the characteristics of CPP, being limited to an action space of only up/down/left/right movement.

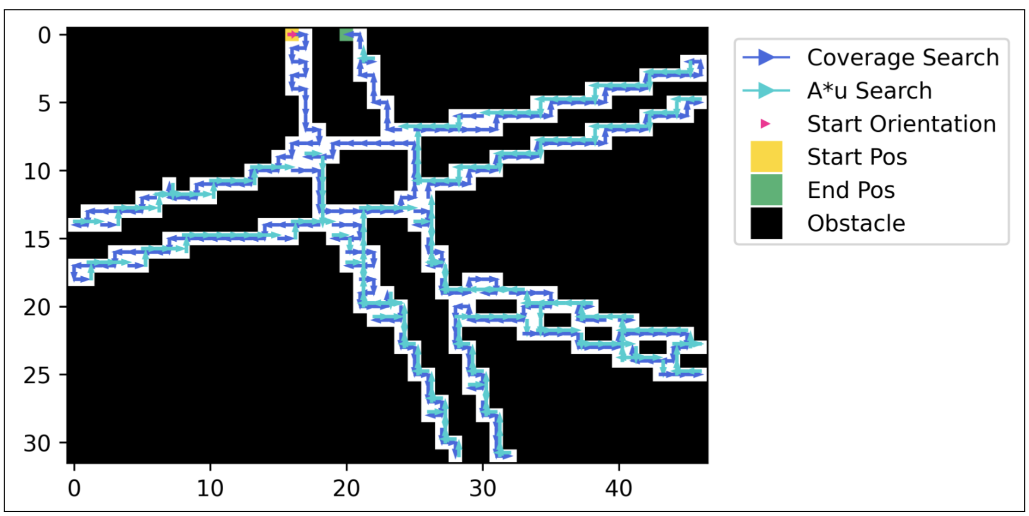


Figure 6.4: Aarhus location ten path planning downsized.

6.3. Discussion and Conclusion

The CPP algorithm plans complete coverage paths for robots from segmented maps. Although CPP is an easy-to-implement algorithm, real-world applications are assumed to require a more advanced solution. These advanced algorithms may not require downsizing, yet this is done with consideration of maintaining recall. The downsizing method introduces some inaccuracy due to floor division and adding nodes to accommodate odd scale factors. It is expected that a robot takes a sequence of coordinates, thus some work must be done to convert the paths for real-world testing. Though the used path planning algorithm is not suitable for real-life applications, it shows that segmented maps are useful for path planning.

7. Discussion

The availability and suitability of geographic data are researched, where coordinate systems, tile systems, and projection are compared. The projection of EPSG:25832 is the most accurate system encountered in the research, though this is based on material from Dataforsyningen. The accuracy of projection methods is not easy to source, thus it can't be ensured that this is the most accurate method available. EPSG:25832 has some amount of error, though it is hard to interpret from the supplied illustration. The size of the projection error is up to 40 cm per km. An error of such size is significant, as it may cause a section of sidewalk to not be defrosted or lead the robot off the path. The developed software does not correct this error, but it should be taken into account for future work.

Tile services generally offer low-resolution tiles, requiring a high zoom level to capture details. Tile services offer faster processing than static services, though this is not a concern for the use case. Generally, static map services are preferred, as these offer higher resolution and cover a larger area, giving a better context compared to non-stitched tiles.

The orthophoto map services offer a variety of data with each their advantage. The source and age of data are not fully disclosed by the providers, which is critical to robotic solutions interacting with the environment. The APIs generally lack documentation about the workings of APIs. The information has been derived from the API requests of their web app or through tests. Working without proper documentation is not only time-consuming but introduces the risk of misunderstanding the data, e.g. the projection might not be as expected.

Navigational maps show poor accuracy; however, the "street-level details" feature offered by Google is being expanded to more cities and will eventually reach Aarhus. It may offer useful data for segmentation, as it offers better accuracy than other navigational map services and includes stairs and other entities that are not available through Dataforsyningen.

The presented segmentation models generally require labeled data for transfer learning to segment sidewalks. Labeling data is a time-consuming task, especially training complex models requires vast amounts of data to achieve generalization. The use of semi- or self-supervised learning hasn't been explored much further than the insights of Mo et. al[21], stating that the accuracy is lower. This could be interesting to explore more thoroughly.

If labeled data was obtained, Tile2Net may be optimized using transfer learning to learn the infrastructure of Aarhus. The segmentation models may also be worth training on administrative maps, as these are simpler maps, not prone to occlusion, and are relatively accurate within city limits.

Dataforsyningen offers spring images, being the most suitable for segmentation, as occlusion from vegetation is minimized. The use of a local tile system makes it unsuitable for use with Tile2Net unless the interface is customized to fit this tile system. The software of Tile2Net is shallowly documented, strict regarding tile systems, and contains errors concerning tile stitching. Stitching errors do not occur in regions within the United States. The only observed difference is Denmark having a coordinate set without a negative longitude, which is a potential cause of error that could be further investigated.

Tile2Net cannot distinguish public and private walking paths, resulting in false positive segments. The filtering methods of contours used in AMS may be applicable to the segments of tile2Net as post-processing, which may increase the performance.

Segment Anything Model performs poorly on satellite images. Though more tests could be performed, based on the poor ability to segment sidewalks, it can be argued that the model is not suitable unless optimized through transfer learning.

AMS demonstrates great recall with some identified sources of errors. The model lacks the ability to filter waterways, railways, vegetated areas, and properties not within a boundary. The data for waterways and railways is available, though it overlaps with sidewalks on bridges and would effectively filter out these. A solution to filter waterways and railways without filtering bridges may increase the precision and reduce the standard deviation. Vegetated areas are not properly segmented, which possibly can be due to incorrect color-filtering and noise-reduction, or the contour filtering does not have the correct thresholds. It is worth noting that AMS is developed for the use of data from Dataforsyningen. If a company wants to offer its services abroad, similar data must be available, which can't be guaranteed.

The contour detection of AMS introduces noise as lines are segmented as areas. It may be argued that fitting vectors to the center of lines may be a more precise approach.

The growing algorithm of OSSA shows potential in the paper, though complications are also presented. The growing algorithm may be able to optimize Tile2Net or AMS with a growing policy more fitting for the general layout of sidewalks in Aarhus. Based on the visual inspections of Tile2Net segmentation results, it can be argued that the scattered false positive areas will have an impact on the performance of the growing algorithm, as the algorithm will possibly try to grow these areas.

The models are evaluated on ground truth maps, manually drawn based on various sources. Some of the sources are used as input to AMS, thus it can be argued that it may contribute to a bias in AMS performance metrics.

The implemented path planning algorithm is a simplified version compared to what is assumed to be necessary for the use case. The robot shall be able to move freely in all directions, instead of being limited to up/down and left/right movement separately. The start position is set as the first sidewalk node, which in reality would be placed at a charging station or storage location. The path planning algorithm visits each node with the robot, requiring the downsizing of maps to a lower resolution. A complete coverage algorithm that considers the robot kinematics to cover multiple nodes at a higher resolution, may give more optimal paths.

The research question *How can geodata be utilized to generate segmented maps of public sidewalks suitable for path planning* is best answered with the use of AMS, as it has the highest performance metrics, though it must be noted that it has a higher standard deviation than Tile2Net. Processing maps for path planning of a defrosting robot depends on the path planning algorithm, yet the used method successfully performed path planning.

8. Conclusion

The aim of this study is to determine how geographical data can be utilized to generate segmented maps of public sidewalks suitable for path planning. As a result, a pre-trained orthophoto segmentation model is evaluated and a custom architecture map segmentation algorithm is developed and evaluated.

It is determined that the use of EPSG:25832 as a coordinate system and projection method is most suitable due to its high accuracy. The projection method does have a significant level of error, yet it is more accurate than EPSG:4326 using Mercator Projection.

The most accurate available segmentation model is found to be Tile2Net, which is limited to the use of a selection of tile services with Web Mercator projection. The software is modified, so Tile2Net accepts other tile services, whereas Google's tile services perform the best. The model performs segmentation of sidewalks with an average precision of 29%, recall of 37%, and accuracy of 91%, with standard deviations of 0.11, 0.13, and 0.03. The use of orthophoto data from spring shows promising abilities to optimize segmentation due to less occluding vegetation from trees; however, the tile system is incompatible with Tile2Net. The performance of Tile2Net decreases when being subject to occluding trees and structures, and when sidewalks are not clearly distinguishable from roads and bicycle lanes.

Navigational maps are researched and show insufficient accuracy for segmentation. Architectural maps are used to construct an OpenCV image-processing-based segmentation model (AMS) using EPSG:25832. AMS performs segmentation with an average precision of 35%, recall of 68%, and accuracy of 89% with standard deviations of 0.20, 0.30, and 0.07. AMS is the best-performing model, though it cannot filter waterways, railways, and structures without boundaries. The model is tested with and without structure filtering, where structure filtering is determined to increase the performance slightly.

A downsizing algorithm for converting segmented maps for path planning is developed, where recall is maintained. The applied path planning algorithm is limited to an action space of moving up/down/left/right separately, which is too simple for proper use for defrosting.

It may be concluded that segmentation of public sidewalks from geodata is possible, though there are grounds for optimization. It is proven that segmented maps can be utilized for path planning to a certain accuracy. This is done using a simple path planning algorithm that requires manual noise removal and drawing paths for crossing roads.

References

[1] Alm. Brand. "Nu kommer alle glatføreskaderne". In: (2022). URL: https://via.ritzau.dk/pressemeddelelse/13664751/nu-kommer-alle-glatforeskaderne?publisherId=13560190.

[2] Retsinformation. *LBK nr 435 af 24/04/2024*. 2024. URL: https://www.retsinformation.dk/eli/lta/2024/435 (visited on 05/10/2024).

[3] Søren Rahbek. *Vinteren kan blive hundedyr: Prisen på saltning og snerydning stiger kraftigt*. 2022. URL: https://www.dr.dk/nyheder/regionale/trekanten/vinteren-kan-blive-hundedyr-prisen-paa-saltning-og-snerydning-stiger (visited on 01/27/2024).

[4] Capra Robotics. *Smart City*. URL: https://capra.ooo/industries/smart-city/ (visited on 01/20/2024).

[5] Søby V. Email correspondance. 2024.

[6] Battersby et al. "Implications of Web Mercator and Its Use in Online Mapping". In: (2014).

[7] The Danish Agency for Data Supply and Infrastructure. "UTM/ETRS89: Den primærekortprojektion i Danmark". In: (2017).

[8] Google. *Legal Notices for Google Maps/Google Earth and Google Maps/Google Earth APIs*. 2021. URL: https://www.google.com/intl/en_ALL/help/legalnotices_maps/#Country-Specific (visited on 04/28/2024).

[9] Mapbox. *Data Sources*. URL: https://www.mapbox.com/about/maps#data-sources (visited on 03/28/2023).

[10] Dataforsyningen. *Forårsbilleder Ortofoto - GeoDanmark*. 2024. URL: https://dataforsyningen.dk/data/981 (visited on 03/15/2024).

[11] Google. *Google Street Maps*. 2024. URL: www.Maps.Google.com (visited on 02/27/2024).

[12] OpenStreetMap. *Copyright and License*. URL: https://www.openstreetmap.org/copyright (visited on 04/07/2023).

[13] Open Street Map. *Open Street map*. 2024. URL: https://www.openstreetmap.org/ (visited on 02/27/2024).

[14] Google. *Explore new street-level details*. URL: https://support.google.com/maps/answer/10311344?hl=en (visited on 03/28/2023).

[15] Gisgeography. *What Is the Web Mercator Projection?* 2023. URL: https://gisgeography.com/web-mercator-projection/ (visited on 03/21/2023).

[16] Dataforsyningen. *Forvaltning og Sagsbehandling*. URL: https://dataforsyningen.dk/data/2680 (visited on 03/07/2024).

[17] Vejdirektoratet. *CVF - Den central vej- of stifortegnelse*. URL: https://cvf.vd.dk/cvf/ (visited on 04/07/2023).

[18] Dataforsyningen. *Matrikkelkortet*. URL: https://dataforsyningen.dk/data/1014 (visited on 05/10/2023).

[19] Kirillov et al. “Segmentation of Occluded Sidewalks in Satellite Images”. In: (2023). URL: https://arxiv.org/abs/2304.02643.

[20] Meta AI. *Segment anything*. URL: https://segment-anything.com/ (visited on 03/03/2023).

[21] Mo et al. “A Survey of Deep Learning Road Extraction Algorithms Using High-Resolution Remote Sensing Images”. In: (2024). URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10933758/#B97-sensors-24-01708.

[22] Liu et al. “LDANet: A Lightweight Dynamic Addition Network for Rural Road Extraction from Remote Sensing Images”. In: (2023). URL: https://www.mdpi.com/2072-4292/15/7/1829.

[23] Hosseini et al. “Mapping the walk: A scalable computer vision approach for generating sidewalk network datasets from aerial imagery”. In: (2023). URL: https://www.sciencedirect.com/science/article/pii/S0198971523000133.

[24] Senlet et al. “Segmentation of Occluded Sidewalks in Satellite Images”. In: (2012). URL: https://ieeexplore.ieee.org/document/6460256.

[25] Open Geospatial Solutions. *segment-geospatial*. URL: https://github.com/opengeos/segment-geospatial (visited on 04/03/2023).

[26] VIDA-NYU. *Data prepare*. URL: https://github.com/VIDA-NYU/tile2net/blob/main/DATA_PREPARE.md (visited on 03/21/2023).

[27] Suzuki et al. “Topological structural analysis of digitized binary images by border following. Computer Vision, Graphics, and Image Processing”. In: (1985), 30(1):32–46.

[28] Renato Fernandes Rodrigues. *coverage-path-planning*. 2022. URL: https://github.com/rodriguesrenato/coverage-path-planning/tree/main?tab=readme-ov-file (visited on 05/15/2023).