Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

School of Computer Science and Statistics

Denoising approaches for data preparation in machine learning

by Chao-Jung Liu

March, 2022

A dissertation submitted in fulfilment
of the requirements for the degree of
Doctor of Philosophy

Declaration

I hereby declare that this thesis is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I agree to deposit this thesis in the University's open access institutional repository or allow the library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

Signed: _____ Date: _____

Acknowledgements

It was a privilege to work and learn from my supervisor Prof. Rozenn Dahyot during the study of my P.hD. This work would not be accomplished without guidance and valuable feedback. Vladimir Krylov and Matej Ulicny work closely as postdoc and co-authors with me in developing deep learning methods, generously sharing profound knowledge and refining the research papers. Also, thank Julie Connelly, our dearest project manager, who helps me deal with all official procedures.

Appreciation also would be given to my PhD peers from V-SENSE people: Tejo Chalasani, Koustav Ghosal, Pierre Matysiak, Matthew Moynihan and Yang Chen. They help me to go through each stage of my PhD and be my eyes to point out the spots where I could not see.

Thanks to family members, their supports are the key to making me accomplish this work. Without your supports, I would not be able to study abroad at the very beginning.

In addition, I would like to thank people from the church groups. They encourage me with genuine words and walk with me in Christ as a church family. Specifically thanks to my mentor Tommy and Christina, who give me free medical consultants and took care of me over these years.

I am also grateful to Adapt Centre, Trinity College Dublin and the people of Ireland for providing all the resources that made this research possible. I also want to thank Nvidia for generously sponsoring me with a powerful GPU that allows me to implement experiments.

And most of all, thank God for his loving grace - whom even coming to the end of this lovingly and gently orchestrates things so that I can run this race. I cannot do this without you. Thank you, praise you and all glory to you forever and ever.

Abstract

Machine-learning models have been recently developed in various computer vision applications, such as image detention, segmentation and so on. Models are often trained based on data that present various levels of accuracy, with typically a better quality of data leading to a higher accuracy of prediction.

This study addresses the importance of training dataset preparation for machine learning applications. We propose to fuse different data modalities for generating our dataset for training a model. However, data streams are not immune from noise and are often desynchronised. Directly employing a noisy training dataset leads to an inaccurate model and predictions. As an alternative to seeking solutions by creating more complex models to tackle this issue, we look instead at improving the quality of the training datasets to train better models.

Our primary contribution is to diminish the noise in the training dataset. We show experimentally that our approach improves performance in applications to building height estimation from single aerial imagery and geotagging of objects from street view images.

For predicting building height from aerial images, our model manages to be within 1.5 meters of the ground truth thanks to a training dataset fusing information from point cloud data and aerial imagery. For geotagging objects, the object of interest can be Geo-localised with higher accuracy by correcting the metadata associated with street view images.

These results suggest that the quality of training examples significantly impacts the result of a model prediction, which leads to a better performance.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Many image analysis applications now rely on machine learning approaches for developing software. In such software, annotated datasets need to be collected for tuning the software to be deployed. ImageNet dataset [70], for example, is a collection of images collected on the web. The creation of such a dataset benefits from the popularity of digital cameras and social media platforms where people can easily capture images and share them online. The Imagenet dataset can then be used to train a machine-learning-based approach for performing large-scale object recognition tasks.

Machine learning-based software is built up through learning thousands of examples, then to be able to perform the task of interest with reasonable accuracy. However, the quality of the training examples directly impacts the machine's performance. A good quality training dataset can be obtained through additional processes to clean or denoise data. This drives computer scientists to explore how to enhance the datasets to improve the accuracy of predictions.

Remote sensing is usually referred to as the process of monitoring the physical characteristics of an area from a satellite or aircraft. The sensing imagery data can be further analyzed using computer vision/machine learning techniques to extract further information that adds to the data's value. Remote sensing is also multidisciplinary research that spans different areas. In Civil Engineering, aerial images can be used for urban planning [103]. In geography, maps can be inferred from aerial imagery, and measurement is computed [122]. For agriculture, the images captured by drones can be used to check the quality of soil and monitor the health of plants [129]. Resilience to extreme climate events such as flooding can also benefit from aerial observation and ground-level imagery [9].

As a result of the technical evolution of digital cameras, very high spatial resolution imagery has become increasingly available, which boosts the research for the applications. Specifically, deep learning techniques have outperformed previous image processing techniques for scene understanding.

By feeding a large amount of training data, the machine can learn tasks, such as image recognition and detection. Deep learning, in some scenarios, can perform even better than a human being in image recognition task [69]. Such milestones convince us to leverage deep learning techniques for extracting geographic information from imagery.

We consider several sources for geolocated image and LiDAR scan datasets. By relying on computer vision techniques, datasets can be leveraged to develop software that can substantially reduce human labour for processing imagery and can be done much more efficiently than manually. In this thesis, our aim is to devise strategies for creating datasets useful for extracting geographic information of value through using imagery. One of our key efforts has been to measure how much better the machine's performance can be improved by learning through better training examples.

1.2 Data Annotation and Preparation

For building a model to understand the image content, lots of data is required to train it. Data availability might not be a problem nowadays, but the annotation associated with the data may not readily be available for the machine to learn.

Hence, the question comes down to the availability of annotation. The annotation can be seen as the additional information that describes the image (see Figure. 1.1). Unfortunately, the annotation usually does not come often with the data itself; therefore, manual imagery labelling is required. It is time-consuming and labour-intensive. Moreover, the bias and errors may be included as a human being is not homogeneous and perfect for doing a labelling job. To resolve this question, we investigated the following applications to avail the data annotation to provide a training dataset.

The use of multi-modality fusion for generating annotation We propose generating annotations using GIS maps to segment 3D point cloud data. The point cloud data can be acquired from different devices, such as LiDAR or, in our case, inferred from multiple images recorded by a drone. In addition, we investigate how contextual information can be fused with 3D data. The fusion of data can potentially be used in AR/VR applications (see Figure. 1.2), for example, for creating a campus tour guidance with VR headset [1]. The application can be developed within the game engine, such as Unity 3D [2].

The height data for the aerial imagery The point cloud data can be used not only for pairing with geographic information but also can be used to associate an elevation map with aerial imagery.

[1] https://youtu.be/n_3fFmszSjo

[2] https://unity.com/

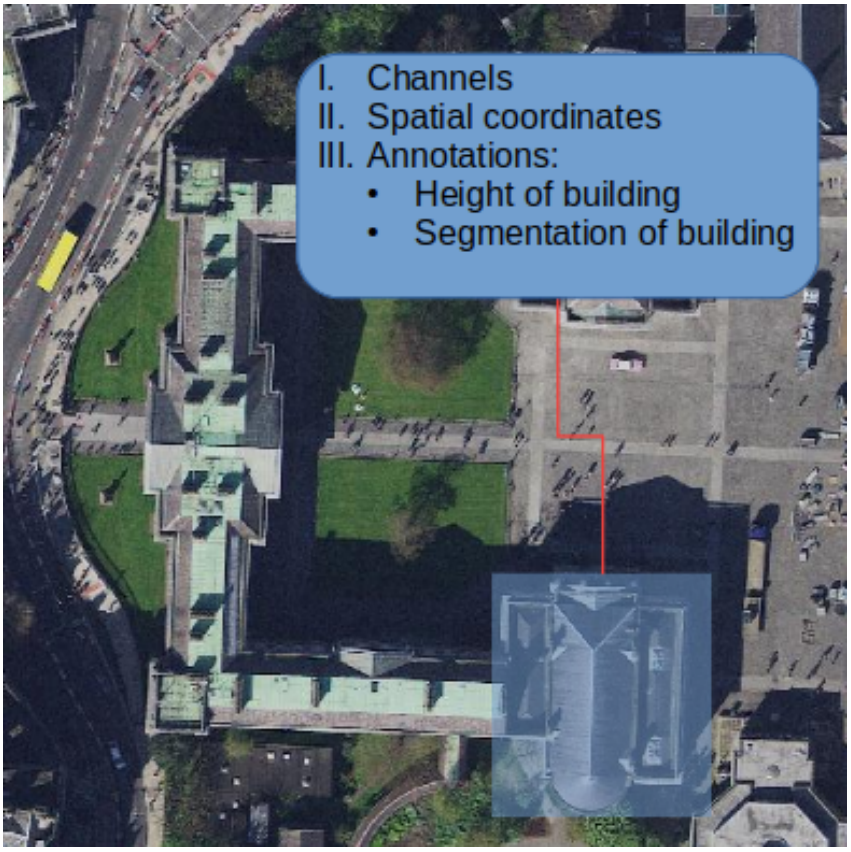


Figure 1.1: Except the visible channels (RGB) [83], the additional layers could come with the image, such as pixel-wise coordinates, semantic segmentation and elevation map.

This association provides a correspondence of height information for every pixel in the aerial image. The goal is to create deep learning to infer elevation from an aerial image. We obtain a high resolution of aerial imagery and align the elevation layer for the corresponding region. This allows to update height information to imagery (see Figure. 1.3). Preparing the dataset is challenging as the data come from different sensors. The accumulating noise from different devices leads to the misalignment of two data sources.

Geotagging of the street view objects The street view imagery is photos captured at regular intervals and covering kilometres of roads with the location. This large amount of street view data can be leveraged to use in many localization and geotagging problems (see Figure. 1.4). The public assets, such as traffic lights, poles or trees, are too small to be well discovered from the aerial view. The street view imagery is complementary to aerial data to discover smaller-sized objects. This application helps to reduce labour in geolocating objects of interest manually. The targeted objects can be updated and geotagged remotely.

Figure 1.2: The contextual information from the 2D map enriches the 3D data by annotating the legend of the building in Unity game engine [82].

1.3 Research Question and Contributions

We investigate the methods for sourcing several modalities for creating training datasets suitable for application in remote sensing and geolocation. We study this problem in the context of three objectives:

- The fusion of GIS information with point cloud data.
- The impact of denoised annotation for deep learning prediction.
- The impact of denoised GPS data for object geotagging from street view.

Publication and contributions

- 3D point cloud segmentation using GIS. C.-J. Liu, V. A. Krylov, R Dahyot. In Irish Machine Vision and Image Processing conference (IMVIP 2018), Ulster University, Northern Ireland

- IM2ELEVATION: Building Height Estimation from Single-View Aerial Imagery C.-J. Liu and V. A. Krylov and P. Kane and G. Kavanagh and R. Dahyot (2020). Remote Sensing, 12 (17).

- Context Aware Object Geotagging. CJ Liu, M Ulicny, M Manzke, R Dahyot. In In Irish Machine Vision and Image Processing conference (IMVIP 2021), Dublin City University, Ireland

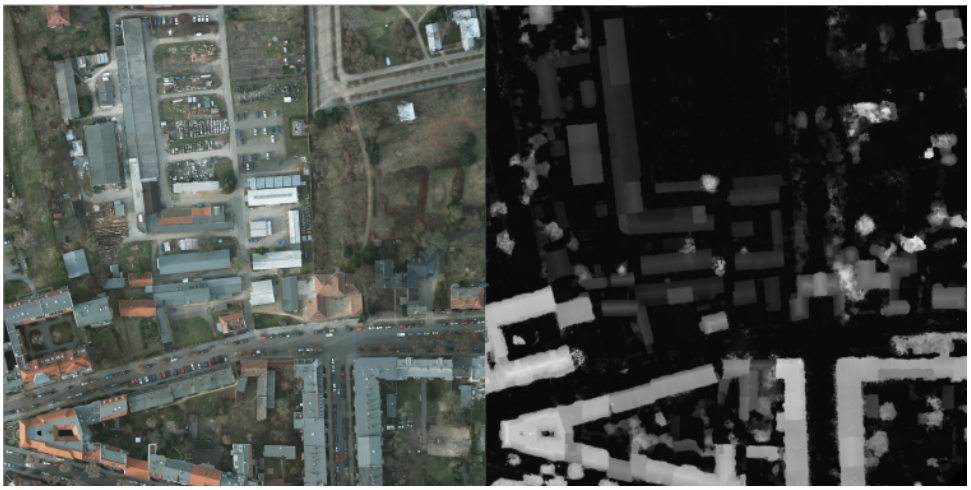


Figure 1.3: The elevation of the building can be predicted from aerial images for detail mapping and analyses of structures. The data comes from the ISPRS Potsdam dataset [3]. The DSM (right) serves as additional information to aerial imagery (left), indicating the elevation information.

1.4 Dissertation Structure

In this thesis, we will focus on generating the annotation and creating a cleaner dataset for remote sensing. We investigate employing different modalities to solve this problem.

- In Chapter 2, we first present the data formats used in applications for inferring geolocated information. Then we discuss the pre-processing methods for generating/denoising the datasets. Finally, we present the related supervised learning architectures in deep learning that we employed to train on the dataset.

- In Chapter 3, we present our initial attempt to align GIS information with point cloud data. In so doing, we study the method of fusing geographic information from a 2D GIS layer with point cloud data, performing the 3D semantic segmentation using information from the GIS source and improving the accuracy of the GIS polygon label. This pipeline generates more meaningful building polygon data from GIS and benefits to machine learning dataset.

- In Chapter 4, different modalities of data are fused to create a larger dataset for training a deep learning pipeline for inferring elevation. To prepare a clean dataset, we deal with the challenge of denoising data using multi-modal registration. Our proposed end-to-end deep learning architecture allows us to map an aerial image to a height map allowing the 3D shape of roofs to be recovered. Such information can potentially be used for computing solar panel energy potential [90].

- In Chapter 5, we propose improvements to Krylov et al.'s pipeline for object geolocation [72].

Figure 1.4: The street view imagery is taken via the mobile app Dioptra[3]. It contains additional information, including the GPS location and the orientation of the camera pose. Object GPS location can be disclosed by giving the extra information from the street view imagery.

We first focus on improving the metadata (e.g. GPS) associated with street view imagery. Second, we propose considering geographic information already available about the scene to refine geolocating objects of interest more accurately. These additional processes improve the accuracy of perdition.

- In Chapter 6, we conclude our main contributions introduced in this thesis and discuss the potential directions to be explored in future work.

[3] https://developers.google.com/maps/documentation/streetview/overview

Chapter 2

Background

In this chapter, we will present the fundamental knowledge that establishes the basis for developing the contributions presented later in this chapter. We will first introduce the geolocated imagery and 3D point cloud data. Then, data preparation is introduced to present the issue of unsynchronised data from different data modalities. Finally, the applications we explored will be introduced. The applications of CNNs and object geotagging are our main focus topics to investigate. We will demonstrate how data preparation impacts these applications in Chapter 4 and 5, respectively.

2.1 Geographic data format for machine learning

2.1.1 Aerial imagery data

There are different types of aerial imagery data (see Figure. 2.2). Normally, they are stored as raster images with tags embedded [1]. The way to capture aerial imagery is varied, which results in different resolutions, and fields of view of the imagery (see Table. 2.1). In remote sensing, images are usually categorized according to the altitude of the aircraft.

Low altitude: A drone survey refers to the use of a drone with a mounting downward-facing camera or LiDAR device to capture aerial data. As the flying height is low, the covering region of each imagery is limited. A region may be photographed several times from different perspectives of views. Some techniques, such as the stereo vision technique [60] or SfM software[2], can be applied to generate a 3D point cloud from multiple 2D aerial images. Using SfM can produce a very detailed elevation map, contour lines, and 3D reconstructions of landmarks or buildings.

However, it is challenging to georeference aerial imagery taken by drone as it is usually captured

[1] https://en.wikipedia.org/wiki/TIFF

[2] https://www.pix4d.com/

in oblique view. For instance, a landmark can be georeferenced with a different location at different viewpoints. This type of aerial imagery can be aligned by estimating the camera pose and then merged into a unified georeferenced image to overcome this issue. This geometric correction is referred to as orthorectification (see Figure. 2.1), and the generated image is called an orthographic image. The orthographic can align with geospatial coordinates, such as the GPS coordinate.

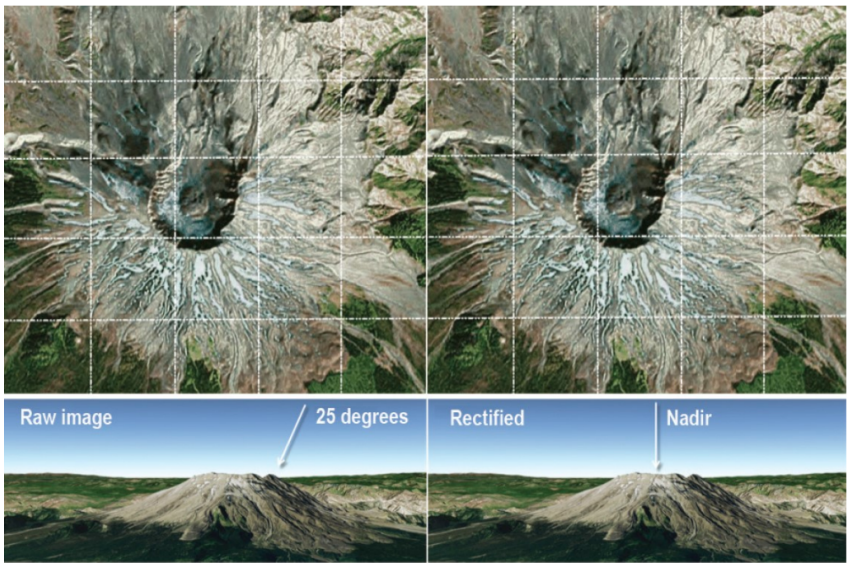


Image source from Esri[3].

Figure 2.1: The process of orthorectification associates the same physical feature appeared in a different view of photographs and generates a straight down view.

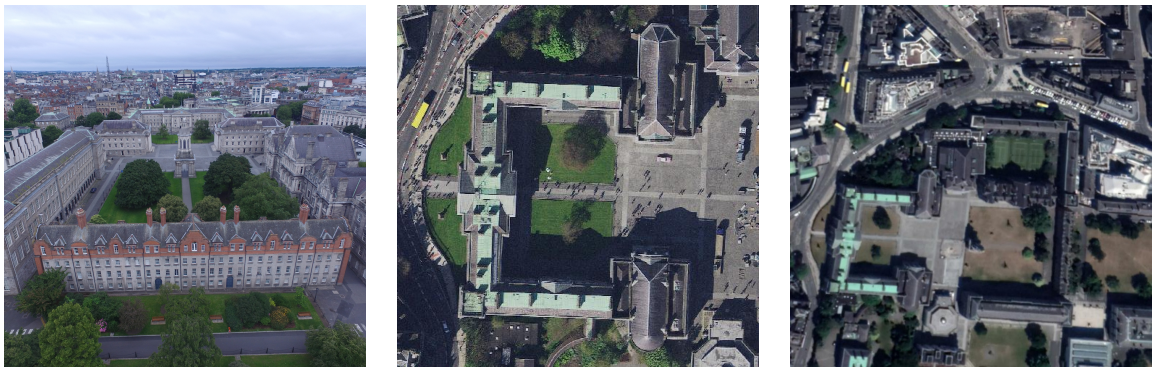
High altitude: An orthophoto is a type of aerial data which is surveyed by aircraft. The survey typically requires a hyperspectral line scanners sensor, such as a pushbroom scanner [44], to capture the aerial imagery. The straight down forward sensor to the ground can produce the orthographic image, which mitigates or removes obliques. Each pixel generated is georeferenced. The correspondence between pixels in a georeferenced image and length in the real world is parameterized by ground sample distance (GSD). For instance, $GSD = 15cm$ means each square pixel in the image is equivalent to an area of 15 cm $\times$ 15 cm in the real world. Therefore, this type of photograph can be directly used as a map on an absolute scale. In addition, orthophoto is normally captured as multispectral bands (RGB + near-infrared). Multispectral bands can be used to detect different characteristics on the earth's surface and also can be used to monitor crop health based on measuring relative reflectance along the wavelengths of these bands [129]. For example, unhealthy crops can be identified by a strongly reflected near-infrared (NIR) signal [45].

[3] https://www.esri.com/about/newsroom/insider/what-is-orthorectified-imagery/

Aerial image	Height	Georeference	GSD	Channels	Captured platform
(a)	Low (100-400 foot)	No	-	RGB	Drone (UAV)
(b)	High (3,000-40,000 foot)	Yes	8-20cm	Multispectral	Aircraft
(c)	Very high (above 500,000 foot)	Yes	10-80cm	Hyperspectral	Satellite

Table 2.1: Comparison of different type of aerial image

Very high altitude: Satellite images (spaceborne photography) refers to the image data captured from space. The data attribute is similar to orthophoto, but the platform flies at a much higher altitude. Therefore, the camera can capture a broader perspective of the earth's surface. For example, Landsat 7 [52] is a platform of satellites collecting seven images at once to generate a 3D image. Moreover, this multispectral imaging allows us to observe cloud-free images of the Earth's landmass.



(a) (b) (c)

Figure 2.2: Aerial images are taken on different platforms. The image (a) was taken by drone [28] with a height of 100-400 ft. This image is highly oblique and allows, for instance, the facade of buildings to be observed. The aerial images (b) and (c) are the orthographic images [83], and only the roof can be observed. Each pixel can be associated with a GPS coordinate. They are captured in different altitudes with different GSD. (b) was taken at the aircraft with 15 cm GSD. (c) was taken by satellite[4]approximately at 70 cm GSD.

[4]https://earth.google.com/web/@53.34391303,-6.25668744

2.1.2 Street view imagery

The street view image data can be regarded as a 2D regular grid with RGB channels. The most common source is Google Street View (GSV). It can be downloaded via the Google API[5]. The metadata comes along with the image encoded in JPG format, including the location of the image captured, the heading, and the image ID. The location is recorded as Global Positioning System (GPS) in WGS 84 [6] coordinate system. The commercial GPS is typically accurate within a 4.9 m (16 ft) radius under the open sky. A GSV is taken with multiple cameras as Google's car moves. Images are stitched, creating a 360 view. It is called a panorama or spherical image (see. Fig.2.3). The centre of the image is the front view of the car.

Figure 2.3: GSV imagery in Dublin city centre. GPS location at (53.3386339, -6.2554441).

Another source of street view imagery we obtained is Mapillary [7]. It is a crowdsourced database of street-level imagery (see Fig.2.4). The platform allows volunteers to upload their recorded data, such as a dash cam and smartphone camera, to their street view imagery database. Unlike GSV, most images are perspective view images, and the GPS position accuracy might be noisier than GSV in the urban area.

Figure 2.4: Mapillary street imagery in Dublin city centre. GPS location at (53.347529, -6.239815).

[5] https://developers.google.com/maps/documentation/javascript/streetview
[6] https://en.wikipedia.org/wiki/World_Geodetic_System
[7] https://www.mapillary.com/

2.1.3 3D data

Point cloud data is the most primitive 3D representation. It contains a set of data points in Cartesian coordinates in three-dimensional space. The point cloud can be created by a collection of images, RGBD camera, and LiDAR (Light Detection and Ranging) device. Each technique holds different advantages/disadvantages in terms of quality, accuracy and completeness. The point cloud data is easy to render because of its simple data structure. However, the lack of topology makes it difficult to perceive the geometrical features.

LiDAR technology measures terrain elevations by shining a pulsed laser from a sensing platform onto a surface and measuring reflected pulses. It is the most common device to collect point clouds from aerial view due to its accuracy and quality. The standard format for storing LiDAR data as points is LAS format[8]. The LAS file contains several fields for each point that can be useful for analysis and display.

Mesh data consists of a set of data vertices (the data points) connected by edges, defining a polygonal shape. This provides a stronger representation for expressing the geometry of a point cloud (see Fig. 2.5). However, generating meshes from point cloud data is quite a challenging process, for which several solutions have been published, such as Marching Cube [86], Poisson [65].

2.1.4 Digital Elevation Model

In GIS, the raster layer is a georeferenced layer that covers a specific area. The smallest unit, also referred to as a cell, contains the coordinate and elevation value. To annotate elevation on aerial images, Digital Elevation Model (DEM) is commonly used as raster layers (See Figure. 2.6). It can be split into Digital Surface Model (DSM) and Digital Terrain Model (DTM).

In a DTM, each cell has a value corresponding to its elevation of the terrain above sea level. A DSM is a raster in which the pixel values represent the elevations above sea level of the ground and all features on it, including all artificial and natural objects. For example, DSM can include buildings and tree heights in the elevation values.

The elevation model can be obtained by converting the point cloud to an image. The image is the projection from the 3D position to the 2D plane. Several reasons exist for converting a 3D point cloud to a 2D elevation model, such as a)less stress on using GPU memory (the memory consumption in training 3D voxel data grows cubically), b)More suitability to apply CNN on 2D grid data, given most of the current CNN architectures are designed for 2D grid data and c)reduce data variance to

[8]https://en.wikipedia.org/wiki/LAS_file_format

[8]https://www.metodoqzero.com/

Figure 2.5: The first figure shows the point cloud data does not have connections between the points. The bottom mesh data shows the connections between the points. The point cloud data and mesh are generated by Pixel 4D with input aerial images from the Trinity drone dataset [28].

Figure 2.6: The creation of digital elevation model from point cloud [137]. Left: point cloud obtained by LiDAR. Middle: DTM describes the elevation of the ground surface. The object above the ground are removed in DTM. Right: DSM records the surface of all objects' elevation above the ground.

simplify the task.

To convert a 3D point cloud to an elevation model, several tools can be used to handle the point cloud data to ease the process. Lastool[9] has the functionalities to render the LiDAR file format in las or laz, and convert 3D point cloud to a 2D elevation model. Alternatively, Pdal[10] is a python package which supports filtering operations. The filter operations create voxels that overlay point cloud data. Each voxel represents a height value, and a height value is determined by computing the median from the number of 3D points within a voxel.

DSM from multi-views

Annotations can also be generated from multi-view aerial imagery. From multi-views reconstruction [60], a disparity map can be generated. A disparity map is a 2-D greyscale image from a rectified stereo pair of images. Each pixel value in the disparity map refers to the displacement between conjugate pixels in the stereo pair image. A larger value represents a near object.

In contrast, the further objects are annotated with a smaller value, which indicates the depth of perception. In the aerial pair of imagery (see Fig. 2.7), the disparity map indicates the object's distance to the camera, which provides the building's footprint in an area.

2.1.5 Open Street Map Data

The geospatial mapping data allows us to query the geographic information in an area. It has been used for various applications, including route planning, location searching, estimated elevation, etc. This type of map service is provided by software companies, such as ESRI[11], or the government agencies, for example, Ordnance Survey Ireland[12]. Due to a limited budget, these authoritative maps

[9] https://rapidlasso.com/lastools/
[10] https://pdal.io/
[11] https://www.esri.com/en-us/hom
[12] https://www.osi.ie/

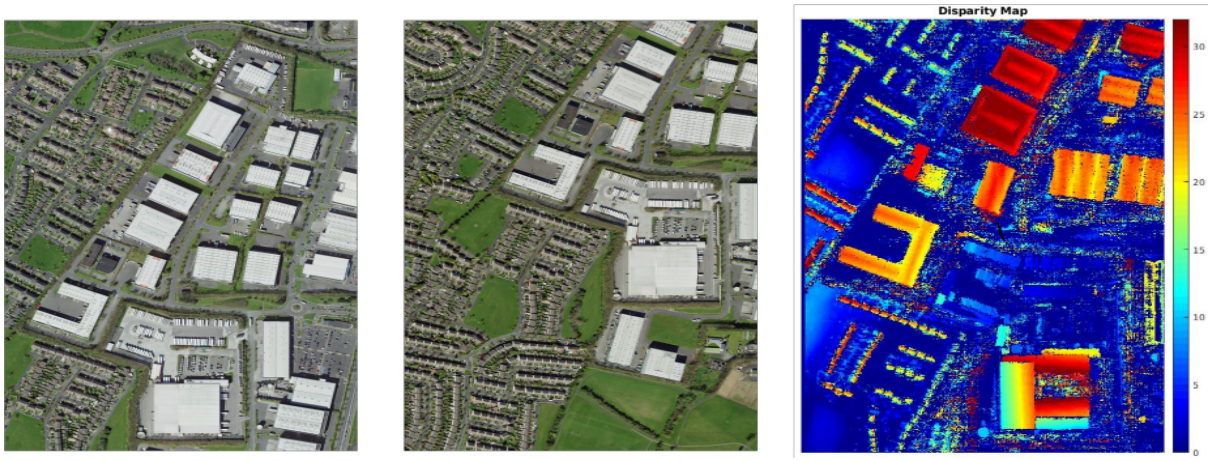


Figure 2.7: The disparity map (right) is generated from the pair of RGB aerial imagery. The dataset is provided by OSM [83].

are not updated on a regular time interval, which may present a lack of completeness and inaccuracy of information.

By contrast, OpenStreetMap (OSM) is the type of Volunteered Geographic Information (VGI) data. VGI is referred to the data provided voluntarily by individuals, which means the data is provided and edited by the public users and is updated regularly. As the OSM is established to encourage the growth and distribution of free geospatial data, it has drawn much attention in the research field of remote sensing in recent years.

The data type in OSM can be categorised into nodes (see Fig. 2.8), ways (polygons and polylines) and relations (logical links). A node can be considered as a data point in a world georeference system, and its coordinate is expressed as latitude and longitude. A way has at least two nodes as a line, for instance, the road from the beginning node to the end node. It can be formed as a polygon if the first node way links to its last one. The building's shape can be depicted using the closed polygon. The increasing popularity of mobile devices with the GPS capacity is encouraging people to geolocate locations through mobile applications, such as OsmAnd[13], Organic Maps[14] and Navit[15]. This large and free geographical information is used in various applications and for research purposes.

2.1.6 Summary

Different modalities of data described above (see table. 2.2) have different attributes. Although the representations are in a different form, they are complementary to each other and enhance the data source for machine learning. We will demonstrate how different data modalities can be introduced

[13] https://wiki.openstreetmap.org/wiki/OsmAnd
[14] https://wiki.openstreetmap.org/wiki/Organic_Maps
[15] https://wiki.openstreetmap.org/wiki/Navit

Data	Format	Key findings
Aerial imagery	Tiff	Aerial view with pixel coordinate
Street view imagery	Png/Jpg	Street view with imagery location
3D data	Laz/Ply	Provide the height information in an area
DEM	Png/Jpg	Derived from 3D data, provide height information in imagery form
OSM	Shp	The vector nodes describe the building polygon

Table 2.2: The table summarises the data available and the key finding information that can be used for machine learning.

into our machine learning application in Section. 2.2.

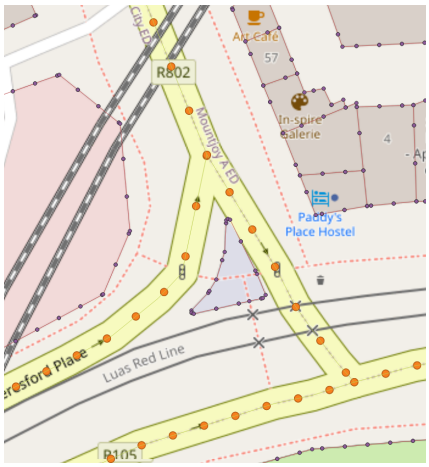


Figure 2.8: The data is imported from OSM. The nodes around the building (purple) form a polygon representing building shapes. The nodes in orange represent the road.

2.2 The Training Data Pre-processing For Machine Learning

Pre-processing the raw data is a necessary step for machine learning algorithms. The main aim of pre-processing is to identify and remove noise from data in order to create a reliable dataset. This improves the data quality for machine learning and enables accurate decision-making.

We investigated different methods to enhance data quality and verify the performance compared to using the original data. The improvements are demonstrated in Chapters 4 and 5 with different applications. In this section, the techniques for pre-processing the training data are introduced in the following subsections.

2.2.1 Building Footprint Overlaying With Aerial Image

In supervised learning, preparing a good dataset is the key to a better model prediction. Ensure the correct annotation for the training source. It usually requires a time-consuming manual labelling process. Hence, introducing another source of data as annotation can potentially ease the labelling process. It saves time and effort to annotate the data from scratch. Fig. 2.9 demonstrates the use of building footprint data to annotate the aerial imagery. For example, Kaiser et al. [64] sourced a large amount of building and road annotation from OSM as a semantic label. They used the label to align with the aerial image and then trained the pair in CNN to extract the road and building from aerial colour imagery; however, if the perfect source that pairs with the training data does not exist, denoising imperfect data is needed. For instance, in remote sensing, misalignment exists between OSM label and aerial imagery (see Figure. 2.10). Even though both sets of data are georeferenced, the noise from the different sensors is difficult to synchronise or remove. The positional accuracy of the OSM data may not be accurate due to GPS limitations, while the optical aerial has relatively accurate georeference. Training with these inaccurate labels leads to poor performance of the model. Hence, it is vital to pre-process these incorrect labels before using them as training examples.

Figure 2.9: Ovelay the building footprint from OSI map data [83] onto aerial image. The preparation of data can be used for supervised machine learning.

The positional errors are independent of instance level, meaning each polygon or line's label has its transformation corresponding to the reference. The misalignment of labels cannot be corrected by a linear transformation (non-rigid transformation).

Vargas-Muñoz et al. [130] pointed out three types of accuracy issues of OSM data in a rural area: i) they are geometrically misaligned; ii) building annotations do not correspond to the updated images (the buildings have been destroyed); iii) some annotations are missing for buildings in the images (the buildings were never annotated). To deal with these issues, they first correct the misalignment by maximising the correlation between annotations and a building probability map. Secondly, they remove the low confidence annotation to optimise their training examples. For those non-annotated buildings, the pre-defined polygon shapes are made to add their annotation.

Similar issues were identified by Chen et al. [32]. They applied perturbations on the building footprint on the noise-free dataset and leveraged CNN to learn the correction, which minimises the misalignment between buildings and their annotations.

Zampieri et al. [138] proposed concatenating the aerial image with its corresponding incorrect annotations as input and leveraged CNN to learn the correct example. The output is the correct version of the aerial with its annotations. A similar approach can be seen in Girard et al. [47]. They employed a multi-scale CNN with a multi-task objective learning process. The intermediate losses inside multiple CNNs from semantic segmentation of building and misalignment of building polygon. They guide the network to learn different features to optimize building polygon and segmentation.

While most research focuses on using machine learning methods to correct the aerial annotation, Liu et al. [82] proposed to automatically correct the building label from OSM using point cloud data and use the corrected OSM 2D label to perform segmentation on a 3D point cloud. This method utilized the intersection of union(IoU) as a function to correct the instance building footprint from OSM. It does not require the training dataset and is faster than optimization in machine learning. However, the 3D point cloud dataset is required to be available as a reference dataset.

2.2.2 Multi-modal registration

The misalignment between the data source and target label can be understood as a registration problem. The registration can be in the same modality or the different one. Regarding the same modality, images are usually captured with the same type of sensor but may be taken from a different viewpoint or at a different time. This results in a misalignment between data sources. This issue can be resolved by matching the features, such as SIFT [87], SURF [17], and then using the Iterative Closest Point (ICP) [142] algorithm to refine the transformation to perform registration.

On the other side, multi-modal registration deals with aligning images which are captured by different sensors by targeting the same area. The images usually have different properties, such as the number of channels, resolution, dynamic range and so on. Therefore, finding the local matching feature to perform registration is difficult. For example, the misalignment happens between the point

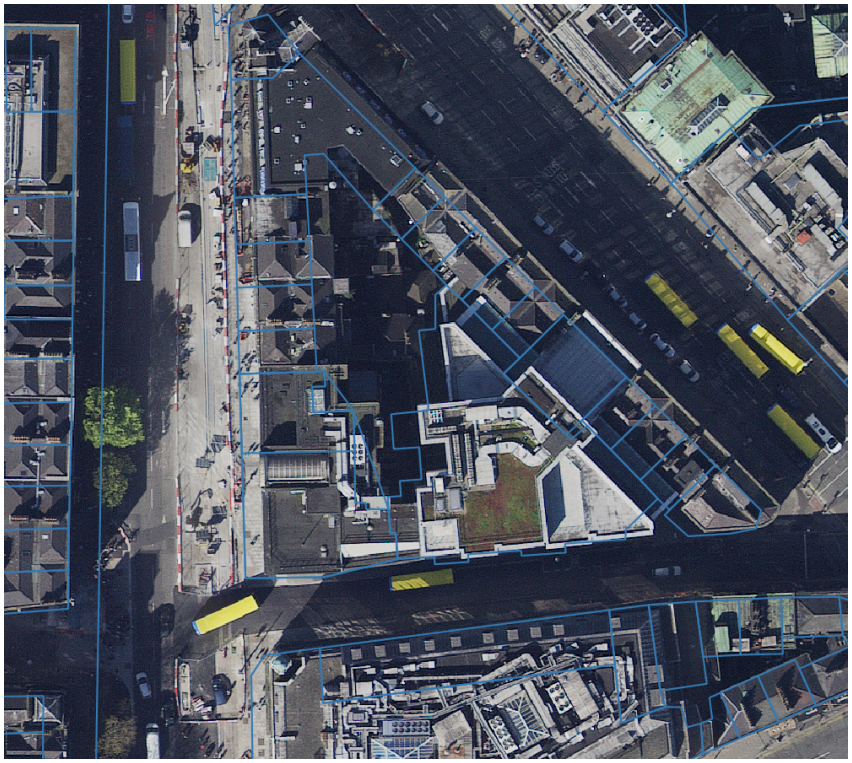


Figure 2.10: The aerial image from OSI [83] misalign the annotations from OSM building. The blue line (footprint) is the annotations from OSM buildings.

cloud data and the aerial imagery (see Figure. 2.11)

In our application, we have encountered multi-modal registration regarding the misalignment between point cloud data and aerial imagery. Directly performing registration from a 2D image to a geo-registered 3D point cloud is difficult as the orthographic aerial image does not have digital camera parameters to perform 2D-3D registration. The more practical way is to rasterise the point cloud to DSM to perform 2D-2D registration (see Fig. 2.12). We break down the methods into feature-based methods and intensity-based methods.

Feature-based methods In feature-based registration methods, different types of features, such as points, linear features and patches, are applied to establish the physical correspondence between different modalities. Habib et al. [55] extracted features of lines and patches. Kwak et al. [74] used the centroid point of the roof. Peng et al. [102] used lines and intersection (corners). Zhang et al. [141] used the objects from the point cloud to form a geometry constraint (e.g polygon). An assumption was made that all point clouds within the geometry constraint in 2.5D would fall into the same corresponding object in the optical imagery. They used the constraint to find the optimal registration solution. This method is similar to Liu et al. [82], who registered polygon OSM to 3D point cloud data. Another novel method from Chen et al. [33] was where they employed deep

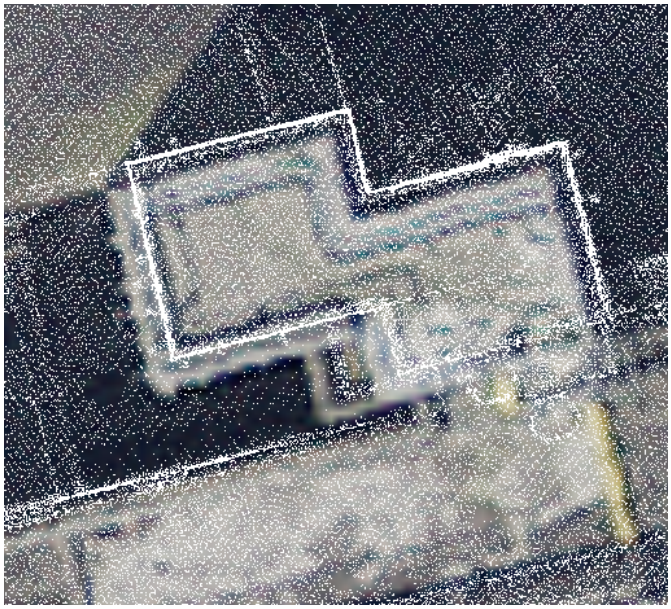


Figure 2.11: The misalignment can be seen from projecting 3D point cloud [75] onto the corresponding image [83] in the same georeference.

learning methods for training registration between polygons and 2-D aerial images. The network would learn the transformation so as to transform polygons to correct positions. However, feature-based methods rely on physical correspondences, which are not always available, especially when using datasets acquired at different times.

Intensity-based methods Information theory has been used extensively for multi-modal registration. The statistical similarity can be measured by mutual information (MI), which utilises the statistical dependencies between the data to be registered and derives similarity measurements [131]. Using MI, Mastin et al. [91] proposed optimising the point cloud's rendering via tuning extrinsic camera parameters. Parmehr et al. [101] registered aerial imagery onto LiDAR point clouds by maximising the combined mutual information from the greyscale-encoded height, return-pulse intensity and optical imagery.

2.2.3 Using SfM Correct GPS Data

Structure from Motion (SfM) is a technique using multiple images to augment 2D detected features to 3D space. The point cloud can be generated through the SfM pipeline (see Fig. 2.13 left). The point cloud can be projected back to 2D to the associated camera views. The re-projection error might happen where the features are misaligned in 2D space. This error can be minimised by fine-tuning each of the camera position. The process is called bundle adjustment [7], which adjusts the camera's position with reference to the 3D features.

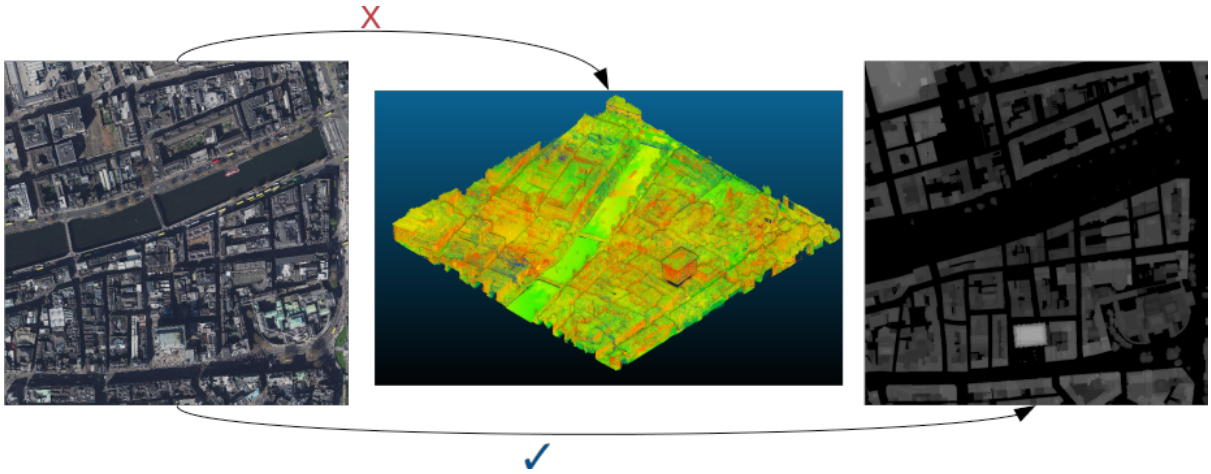


Figure 2.12: Directly registering from 2D [83] to 3D [75] in different modalities of data is difficult. It is easier to transform the point cloud to DSM and perform 2D-2D multi-modal registration.

Generally, the more images we feed into the system from an area, the better the results are. An SfM-corrected sequence is shown in Fig. 2.13 right. The original image locations(green dots) deviate off the side of the road. The red dots are the corrected locations after SfM, which are aligned with the roads.

2.2.4 Summary

We demonstrate the data preparation in this section . Although the data describes the same location, there is misalignment between different data sources. The registration gets involved in correcting the misalignment to generate a cleaner dataset.

section Supervised Learning with CNNs In this section, we discuss the topics of deep learning-based applications related to our work, including the tasks of object detection, semantic segmentation, and depth estimation. In addition, we will briefly talk about deep learning evolution and its architecture in the Section 2.3. We will talk about how the CNNs can be applied in different tasks in computer vision applications which motivate our work in Section 2.3.2.

2.3 From Machine Learning to Deep Learning in Computer Vision

Most tasks involve computer vision attempting to encode the image to a latent space. This encoding process allows a machine to access an image's local features. Those features keep the most important pattern to describe information from images. For instance, Principal component analysis (PCA) [133] reduces the dimensionality of image data. It works on identifying the importance of features through computing the covariance matrix of an image and breaking the matrix down to eigenvectors and

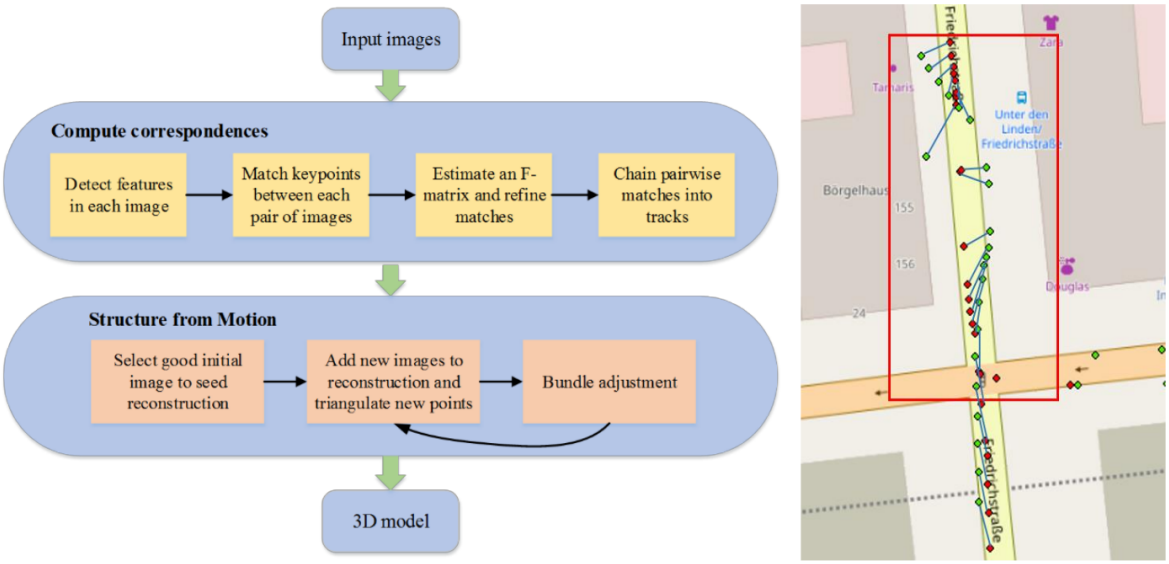


Figure 2.13: Left: SfM pipeline. Right: The result of the corrected camera view position(red dots) versus the original position from GPS data(green dots). The proposed pipeline [139] reduces the GPS noise, providing a more precise camera location.

eigenvalues. They can describe the local geometric characteristics of pixel intensity. The handcrafted image descriptors, such as SIFT [87], SURF [17], are also used to detect the image's features. The common attribute of these features is based on the representation of local geometry. Therefore, these features may not generalise well to unseen or new distributions.

Encoding images with handcrafted features may not be enough to generalise the data source. Simply using the features with linear estimators, such as support vector machine [124], is limited in their ability to learn the mapping from source to target domain.

The neural network, known as multilayer perceptron (MLP), is inspired by the human nervous system, which consists of neurons (nodes) and synapses (connection). It can be represented by a directed acyclic graph that includes multiple layers. However, the number of parameters in MLP grows exponentially with the input size. The memory is easily used for high-dimensional data, such as images. With the rise of image data on social media and the high-resolution images from digital cameras, we need a scalable solution to apply image data to develop real-world applications. Convolution Neural Network (CNN) is a computationally efficient and scalable version of neural networks. CNN can handle this efficiently by using the local receptive field and weight sharing to achieve scalability and high performance on image data. The pooling layer is introduced to reduce the feature dimension and lower computational efforts. Additionally, it has several other benefits, such as translation invariance and hierarchical representation to handle the global features of the whole image.

2.3.1 The history of CNN architectures

AlexNet In 2012, Krizevsky et al. [70] proposed AlexNet. The network deals with the image recognition task, which won the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) [38]. They reached a very high accuracy by defeating the first runner-up by 10.8% - a significant margin of error. It consists of eight layers, five convolutional layers, and three fully connected layers. They also introduced max-pooling and ReLU activations, two key concepts widely used in CNNs. In addition, they were the first to introduce training CNN in multiple GPUs(see Figure 2.14). The authors showed that depth was a crucial factor that affected the efficiency of CNNs.

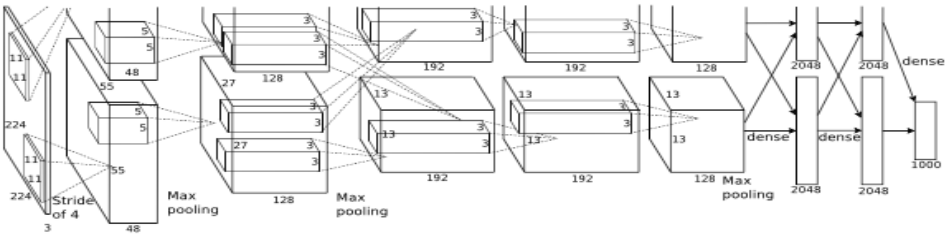


Figure 2.14: AlexNet architecture [38].

VGG-Net In 2014, Simonyan et al. [119] proposed VGG-Net(see Figure 2.15) to focus on the depth of neural network. They used a smaller receptive field (3x3). The network started with the image dimension of 224x224x3, resulting in a thousand channels (one for each class). They demonstrated that having a smaller receptive field and adding more layers made it learn hierarchical representations. Furthermore, instead of splitting the network into two, such as in AlexNet, which computes the weight of different GPUs, they combined all parameters into one and computed on one GPU, making the architecture more compact. It was the first runner-up in ILSVRC 2014.

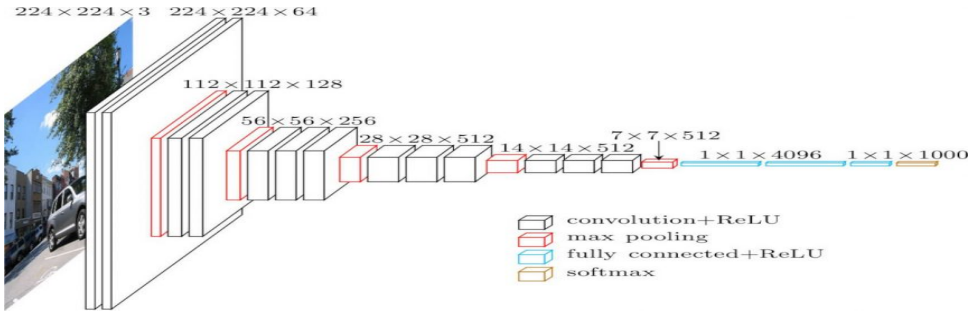


Figure 2.15: VGG-Net architecture [119].

ResNet Since deeper networks contain more layers, they have an advantage over shallower networks by obtaining higher levels of abstraction of the convolutional layer. However, due to many layers,

deeper networks encounter a problem of vanishing gradient while training. It causes the network to perform worse than a shallow network.

Residual network architecture [57] was created to resolve this problem. It creates a residual block (see. Figure 2.16) that copies the input value from the previous layer to merge the output from the non-linear function, allowing earlier layers to access the gradient signal from later layers. In other words, skipping the operations on the non-linear function allows earlier layers to gain a stronger gradient.

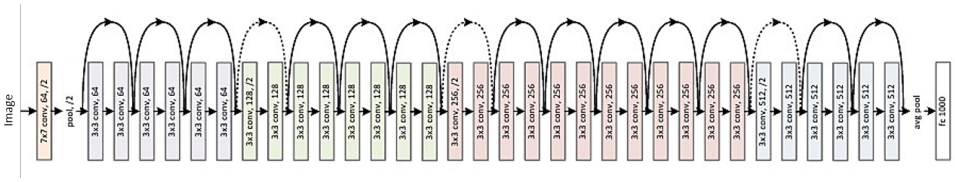


Figure 2.16: ResNet architecture [57].

DenseNet Huang et al. [63] proposed DenseNet, which improved ResNet using denser connections but with fewer parameters. The DenseNet architecture uses the residual mechanism to its maximum by making every layer densely connected to its subsequent layers (see. Figure 2.17).

Compared to ResNet, the advantages of DenseNet are: 1) DenseNet has a stronger gradient while training because the error signal can be easily propagated to earlier layers more directly. 2) While in ResNet, information from the last layer is passed using element-wise addition. DenseNet combines information from all the previous layers using concatenation. They reduced the output channel to reach a better computational efficiency.

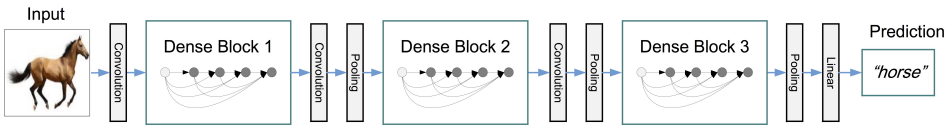


Figure 2.17: DenseNet architecture [63].

From the AlexNet to DenseNet, the model gradually became more complex. We can see the intra-convolutional layer sharing from one layer to another layer to gain the ability to encode the image into latent space. The distinct latent variables can be connected to an output layer which will return the prediction of the image's class.

Fully convolutional network On the other hand, in Fully Convolutional Network (FCN) the fully connected classification layer is replaced by a decoder which maps the latent features to reconstruct the original or a transformed image. This type of network is referred to as fully-convolutional network

(see Figure. 2.18). This network architecture is used for other computer vision tasks (see Figure. 2.20), such as semantic segmentation or depth estimation.

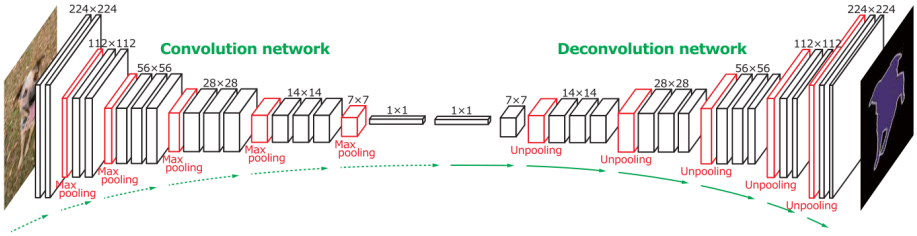


Figure 2.18: Fully-convolutional network architecture [99].

Region proposal based network Different from normal CNNs, the region proposal CNN (RCNN) creates the number of regions of interest (see Fig. 2.19). Those regions are resized into the same size image and fed into a normal CNN.

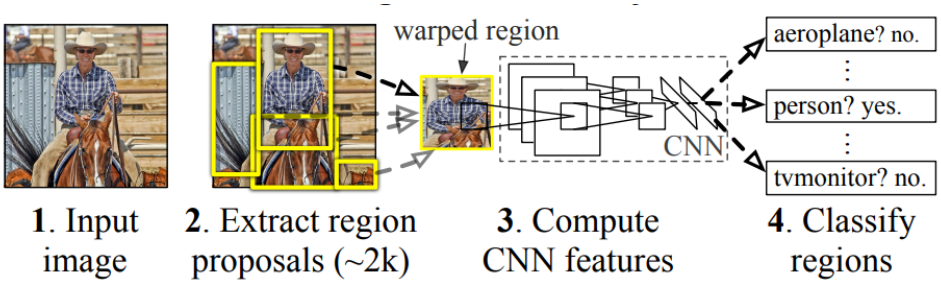
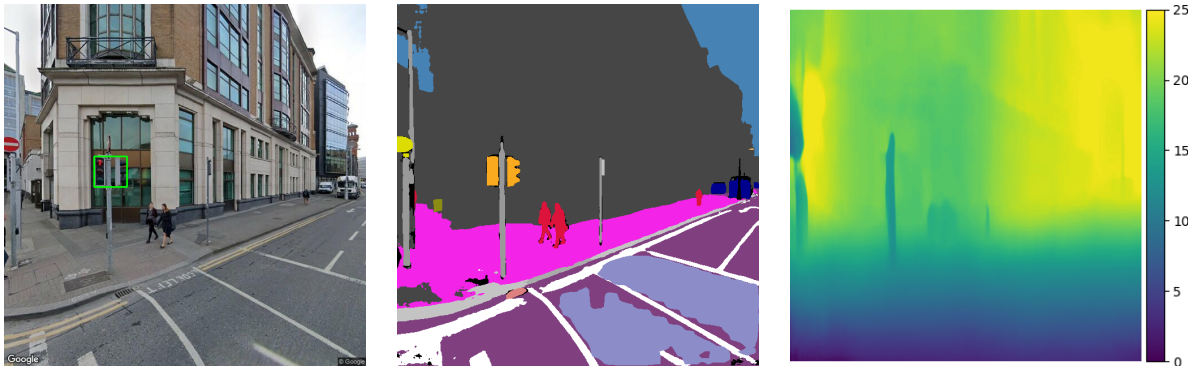


Figure 2.19: Region proposal CNN [49].

2.3.2 Applications of street view object detection

Detection

Girshick et al. [49] proposed R-CNN to localise and segment objects. R-CNN requires gathering plenty of regions of interest to be accurate, which causes poor performance during model inference. In their following work [48] improve R-CNN where instead of extracting features from each proposal, the fast R-CNN extracts features as an entire image by fusing a number of the ROIs. Ren et al. [109] proposed faster R-CNN real-time performance by using a separate network to predict the region proposals. Kaiming [56] proposed Mask-R-CNN, which is also based on fast-R-CNN architecture. They added a few convolutional layers to generate an object mask.



Detection Semantic segmentation Depth estimation

Figure 2.20: The street view was captured from Dublin city centre at GPS location (53.346 , -6.255). Left: the detection task [34] detects the traffic light in the image. Middle: the semantic segmentation [105] classifies every pixel in the scene. Each colour scheme represents different objects. Left: depth estimation [50] deals with the relative distance to camera in the image.

Semantic Segmentation

Semantic labelling is a task in computer vision which makes a decision based on the category of every pixel. This would assign each pixel with a pre-defined class. The class label is chosen by computing the combination of texton(texture), shape and appearance. TextonBoost [118] and texton forests [117] have achieved compelling results. It predicts a random pixel associated with a potential function that computes the cost of that pixel taking the semantic label. The output mask would be the same size as the original image. The Deep Neural Network is the current state-of-the-art framework to make classification of each pixel. CNN extracts features both in contextual information and local feature of an image.

Badrinarayanan et al. [15] proposed the SegNet that combines convolution and deconvolution layers to make the pixel-wise prediction. They utilise max pooling vertices to reference the deconvolution layer to reduce the weight parameters.

Depth estimation from single image

Single depth estimation is a challenging problem. It has scaling ambiguity, ill-pose as well as occlusion issues. However, human beings can estimate the depth from visual experience by using the naked eye. The same idea can be applied to machine learning by feeding the machine to learn many images with depth annotation.

More recently DCNNs-based models have been actively investigated [40, 84, 134, 85, 78, 89]. These methods attempt to directly predict the pixel-wise depth estimation in an image using models

that have been trained on a vast number of ground truth depth data.

The encoder-decoder with skip connection framework has been a common trend in CNN architecture [70, 57, 63, 61, 89] and has been widely used both in single-depth estimation and semantic segmentation.

2.4 Object geotagging from street view images

It is labour-intensive and costly to monitor the public assets in an urban area. The maintenance requires a group of people regularly to identify objects and measure geolocations to feed its inventory database. This repetitive task can be aided by using software applications to automate the process.

SLAM/SfM [36][114] techniques are able to reconstruct 3D scene from images. These techniques can be used to recover the static scene and locate the public assets, such as traffic lights/signs. Other than that, the LiDAR technology can be also leveraged to generate the map with the high precision [115]. For recording public assets' location, it is not necessary to use LiDAR scanner to have Centimetre-level positioning accuracy. The optical street view data would be an alternative solution for updating the in situ inventory and its geolocation.

Updating inventory and geolocation of street view objects is a highly relevant task. In fact, Mapillary is currently encouraging their users to provide such information via their mobile phone's GPS location near the street objects to the databases manually. However, exploring a vast amount of street view images allows us to automatically discover such objects remotely in a much more efficient fashion.

To locate an object of interest, it consists of 3 fundamental tasks: 1) Where is the object targeted in the image? 2) Where is the camera position in the world coordinate? 3) The distance from the camera optical centre to the object of interest. Each task may affect the accuracy of the object's geolocation. There are several methods which are actively being researched to solve the problem, which we have categorized in the following sections.

Probabilistic model based

Wegner et al. [132] exploited different modalities of data to determine the tree's location. They used multi-street views along with aerial views to locate trees. Except using information from imagery, the prior from spatial context is imposed on the distance between neighbouring objects. For example, the gap between two trees is regulated to plant in close proximity. Moreover, they used the prior from Google Maps to push the detected trees away from the middle of the road. The assumption is that the tree was planted alongside the road. Krylov et al. [73] proposed to employ the monocular depth estimation, object detection and triangulation from pair of views to estimate the position of

static objects. The approximate knowledge about the camera-to-object distances is transformed into the ray with the specified length. The MRF model is applied to decide the positive ray intersection, followed by clustering to estimate the object's geolocation. A similar approach proposed by Weixing et al. [140]. They purely used triangulation without single depth estimation to locate the targeted object. More than two views are used to detect the same object across camera views. However, the performance is much worse than Krylov's proposal. The accuracy of the predicted location is 2-4 metres away from the reference point in Krylov's method, while Weixing's method only reached the accuracy at less than 10 metres.

CNN based

In the multi-view scenario, similar objects may be identified as the same object in different views. Nassar et al. [96, 97, 31] used the constraint from a given camera pose to re-identify the same object that appears in two views. They encode the camera pose information alongside RGB channels as the similarities can be derived not only from visual features, but geometric features, and jointly learning these features in a single end-to-end fashion increases the performance in the two-view scenario. The same author extended the idea by constructing a Graph Neural Network (GNN [67]) to identify the relationship between the detected position across different views.

SfM based

The camera pose from image metadata might be noisy as GPS accuracy is within a 5 metre radius under the open sky, and the accuracy worsens near buildings, bridges, and trees. Purely relying on the position affects the accuracy in predicting the object's position. A sequence of images is able to match with local features, and compute the geometry relationship with the matching views. This can reduce the noise from the camera pose. Hebbalaguppe. et al. [58] proposed to use two views with bounding box detection to locate the object of interest. A similar approach was introduced by Zhang et al. [139] where they employed SfM to correct camera pose from image metadata. Semantic segmentation was applied to generate the graph to depict the structure of the scene. To localise the objects of interests, the topological binary tree was used to encode the several rules of the relative position in the scene based on their assumptions.

2.5 Summary

In this chapter, we investigate a variety of data in section. 2.1. In 2.2, We discover that different data is not perfectly registered with each other due to the noise. Therefore, the registration is needed

to create a cleaner dataset. To test the dataset, we study the deep learning methods in computer vision in 2.2.4 and its application 2.4 to verify our data preparation.

Chapter 3

Point cloud segmentation using GIS

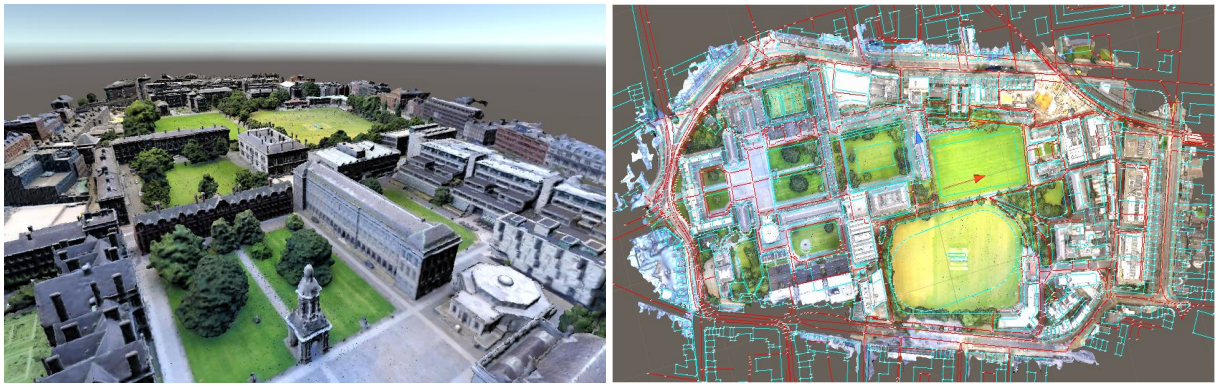
We propose an approach to perform semantic segmentation of 3D point cloud data by importing the geographic information from a 2D GIS layer (Open Street Map). The proposed automatic procedure identifies meaningful units such as buildings and adjusts their locations to achieve best fit between the GIS polygonal perimeters and the point cloud. Our processing pipeline is presented and illustrated by segmenting point cloud data of Trinity College Dublin (Ireland) campus constructed from optical imagery collected by a drone.

3.1 Introduction

Up-to-date, accurate 2D and 3D maps are growing increasingly important for localisation and navigation employed by both humans and machines. Various technologies and data collection modalities are available nowadays to capture and encode a digital twin of the world, such as LiDAR [75] and drone imagery [28]. These data streams synchronise with the features of the physical worlds in the digital representation, enabling the necessary data-driven analytics and machine learning purposes. Such digital twins can be seamlessly manipulated and visualised with the help of game engines and used in applications, such as education (e.g., driving simulators) and entertainment (e.g., virtual visits and gaming [54]). More recently, these virtual environments have also found applications in providing valuable labelled data for training machines using data-driven artificial intelligence. For instance, project Airsim [1] uses the Unreal game engine to provide training data for autonomous drones and cars.

We aim to label automatically unstructured geolocated 3D point cloud data. For this purpose, we propose to register heterogeneous sources of information: the semantic information provided by Open Street Map (OSM) and 3D point clouds covering the same geographic area (Fig. 3.1). We

[1] https://github.com/Microsoft/AirSim



(a) Trinity 3D model dataset [28].

(b) Overlay OSM data onto game terrain.

Figure 3.1: 3D point cloud of Trinity College Dublin campus, 2015.

show that the registration of these two sources of information allows one to segment the point cloud into useful semantic units with arbitrary geometrical shapes. After introducing some related work (Sec. 3.2), our approach is presented in Section 3.3.

3.2 Related works

Structure from motion techniques have been widely used to reconstruct large-scale scenes and image geo-location. Practically, it has become a cheaper alternative to costly high-end LiDAR technology. Structure from motion relies on keypoint detection, image matching, bundle adjustment and generation of dense point clouds. These points can then be condensed into meshes and be presented in the game engine as terrain (cf. Fig. 3.1a). For instance, Agarwal et al. [8] use a large collection of pictures harvested from the web to create a sparse point cloud of a city. Bódis-Szomorú et al. [22] proposed to reconstruct a mesh of a city area by registering and fusing two point clouds, one airborne and the other generated from street view and aerial images to reconstruct the city view, and then generate the mesh by using fusion techniques. Rumpler et al. [111] register OSM data for retrieving the information of outlines of buildings in the form of 2D vectors, and use them with satellite pictures to generate 3D mesh building. Bulbul and Dahyot [27] reconstruct a cubical model of a city using OSM geolocated building footprints and heights and associate textures extracted from Google Street View imagery to recover the building facade. The resulting 3D city model is then used in a game engine to visualise social media activity in the area, using pictures posted on social media platforms.

There is a large body of work for image region labelling into generic categories such as cars, trees or buildings. For instance, Tighe and Lazebnik [127] proposed an image parsing method to segment the region of images by trained descriptors. The semantic labels are classified by predefined class for geometric/semantic context. The semantic labelling work is conducted through all of the overlapping

images for reconstruction, which is very time-consuming. Having a 3D mesh generated multiple view imagery, Riemensschneider et al. [110] improve computation complexity and accuracy by predicting the best image view for clustering and propagating the labels to the mesh. It is shown to be more efficient than clustering all images and merging labels for a same object in the scene. Tchapmi et al. [126] label raw point clouds using neural networks with a Conditional Random Field to predict the class of each point in the cloud. Kaiser et al. [64] segment the 2D aerial imagery into an object by using the data from OSM, transforming geographic coordinates to local pixel coordinates, and use neural network architecture to refine the boundary to make finer-grained labelling for objects. Becker et al. [18] proposed to classify photogrammetric 3D point clouds into specific classes (e.g. road, building, vegetation).

Krylov et al. [73] proposed to geolocate street furniture from Google Street View (GSV) imagery using fully convolutional neural networks for object segmentation and distance estimation followed by a Markov Random Field to coherently geolocate individual objects in images. This was extended in [71] to the fusion of street-level imagery and LiDAR data to object detection at increased spatial accuracy. Similarly, Branson et al. [25] use GSV images together with optical satellite imagery for cataloguing street trees.

3.3 Geolocated data registration

Our approach is demonstrated using the data for Trinity College Dublin Campus recorded using a drone in 2017, which is available open source [28]. Our methodology employs the 3D point cloud generated from drone imagery (Sec. 3.3.1). We employ OSM data as GIS source of information [111, 64, 27].

3.3.1 3D point clouds generated from drone imagery

The Trinity campus mesh was generated from optical imagery captured by a drone [29] in 2017. Pix4D software[2] has been employed to process the images to generate the geolocated point cloud using structure from motion techniques.

3.3.2 OSM data structure and parsing

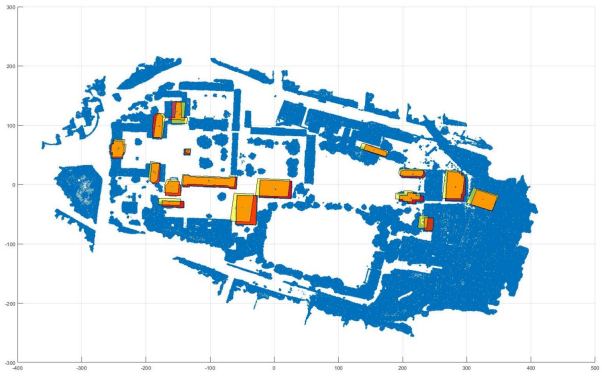
To acquire geo-location and semantic label data, we employ the OSM API that allows one to extract information about a specific geographic area from their database in XML format. We use this interface to collect semantic information about the Trinity campus. The XML dump file contains all three types

[2]https://pix4d.com/

of OSM attributes: ways, nodes and relations. Ways are used to encode polygon-shaped areas like buildings, roads, etc. A way contains semantic labels (tags) and references to corresponding nodes. Each node contains the geo-location (longitude and latitude) of an individual point on the map (e.g., corner of a building). Relations are used to model local logical or geographic relationships between objects. Using the name of any individual building, we can search among all the way attributes to find all the nodes establishing their location inside the Trinity campus.

3.3.3 Mercator projection

OSM uses the WGS84 spatial reference system. This is the reference system also used by GPS. The corresponding coordinates are referred to as longitude and latitude. WGS84 models Earth as a spheroid. The real shape is, however, ellipsoidal, i.e. flat at both poles. To resolve this inconsistency, the Mercator projection[3] is employed. The point cloud data we obtain is not georeferenced data. In order to georeference the data, an OSM map is used to introduce the geo-location of buildings. The Mercator projection is applied to ensure a more accurate georeference coordinate in the local area.



(a) Intersection (orange) between OSM building positions (yellow) and 3d point cloud ground truth locations (red).

	IoU	Distance (meters)
Average	0.567	5.594
Median	0.607	4.871

(b) Distance between centroids defined by OSM and 3d point cloud across 16 buildings.

Figure 3.2: Fit between point cloud data and OSM GIS data after global registration.

3.3.4 user defined correspondences between heterogeneous data streams

To align the reference location from the Mercator projection to the 3D model, we manually selected 18 correspondences (control points) between the two media streams to estimate an affine transformation matrix handling translation, rotation and scale with the Least Squares. The 2D affine transformation has six unknown parameters. We purposely found more ground control points than required (minimum

[3]https://wiki.openstreetmap.org/wiki/Mercator

2 points) to stabilise the Least Squares solver, and we discovered that the solution stabilised after 18 points of input.

The centre of Trinity campus model (0,0)is on the Museum building, has OSM coordinate (53.34380,-6.25532) and Mercator projection coordinate(-696339.0371489801,7012543.77625507).

3.3.5 Evaluation of global affine registration with user-defined correspondences

To evaluate how accurate the geo-locations of nodes from OSM fit the considered point cloud dataset, we report assessment on 16 buildings across the campus. We also manually label the point cloud by photo-interpretation to have a ground truth of semantic units corresponding to these 16 buildings. Each building is defined by its perimeter, i.e. a polygon shape on an x-y plane in an orthographic view (see red polygons in Fig. 3.2a). The information about building locations coming from OSM is reported as yellow polygons in Fig. 3.2a. We compare these two polygons as 2D shapes (discarding the third dimension of data) to assess the correspondence via overlap using *Intersection of Unions* (IoU) [35]. The IoU is a real number between 0 to 1, where higher values mean better fit (equivalently, stronger overlap). We compute the shape's centroid points and use these to find the distance between polygons.

As can be seen (Fig. 3.2a), there is a substantial misalignment between geo-locations reported by OSM and 3d point cloud. This is mainly because one single affine transformation does not capture well the deformation between the two streams of information. Furthermore, the location information in OSM is crowd-sourced and is not always of the highest spatial accuracy. Figure 3.2b reports the averages and median values for IoU and the distance between OSM and 3d point cloud polygons' centroids. Fig. 3.3 demonstrates the IoU for each of building and it reflects to the misalignment for each of building.

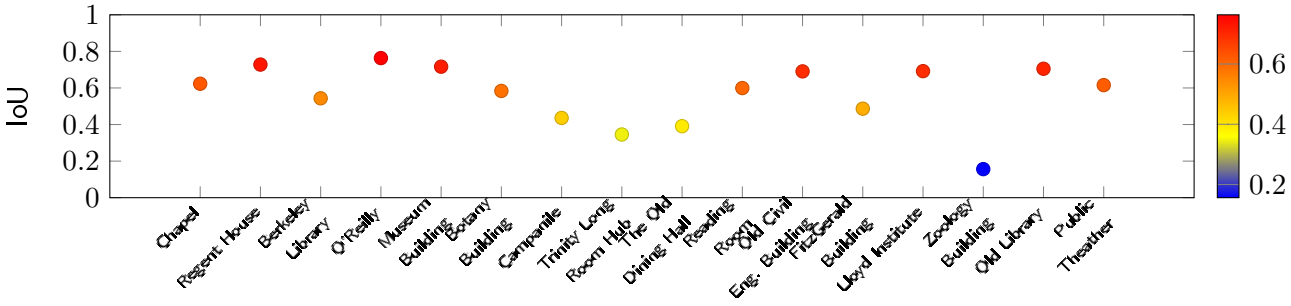
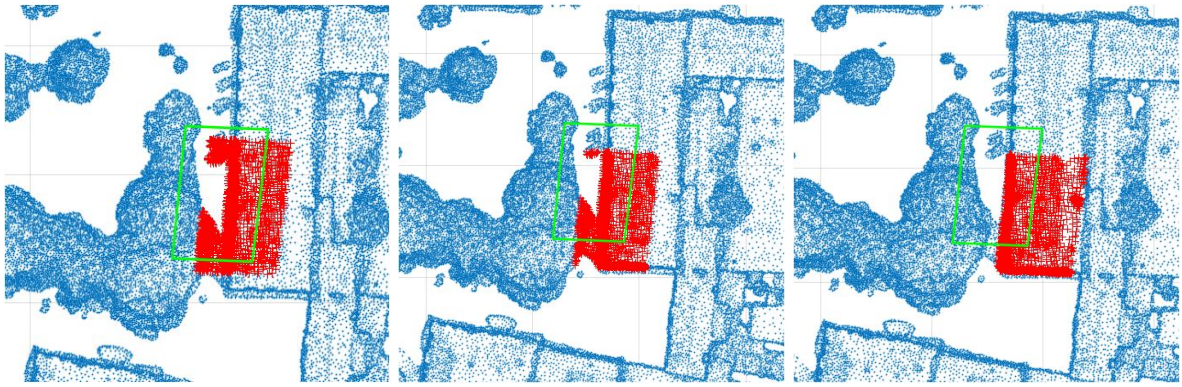
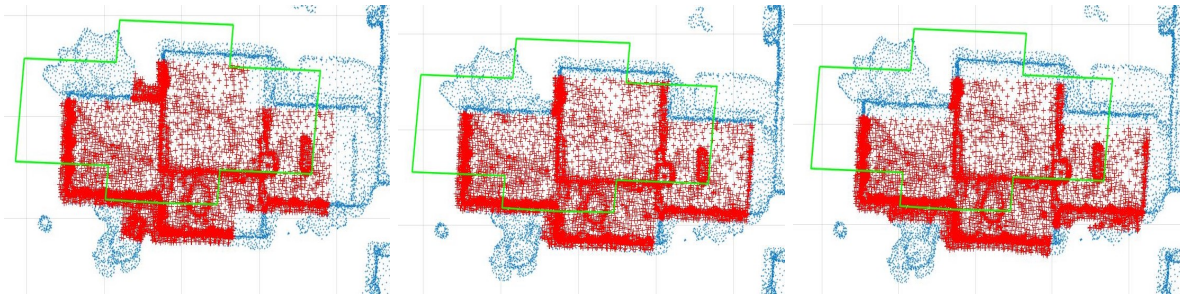


Figure 3.3: IoU per building after global affine registration.



(a) Zoology bld., $r_3 = 5.5m$ (b) Zoology bld., $r_2 = 8m$ (c) Zoology bld., $r_1 = 12m$



(d) FitzGerald bld., $r_3 = 5.5m$ (e) FitzGerald bld., $r_2 = 8m$ (f) FitzGerald bld., $r_1 = 12m$

Figure 3.4: 2D point cloud after ground removal (blue points) and the segmentation (red points) obtained by the proposed OSM adjustment procedure for various search radii r exemplified on Zoology and FitzGerald buildings. Green polygons represent the original OSM building positions.

3.4 Adjustment of OSM locations

Alongside the 16 buildings, our 3d point cloud comprises other points describing ground, trees and other buildings. We first remove the ground points by elevation surface thresholding using CloudCompare (available at `https://www.danielgm.net/cc/`). We then discard the third coordinate data in the 3d point cloud (elevation above the reference level), transforming it into a 2d point cloud D_{2d}.

We employ IoU as the main metric to assess the quality of fit between the data and ground truth locations. In order to improve IoU for buildings, we propose to estimate the optimal translation from OSM-defined positions towards 3d point cloud clusters. We assume that the spatial orientation (rotation) and scaling are correct in OSM. Please note that we made this assumption not because the scale and rotation are perfect but because we aimed to find the polygon positional errors.

In other words, to identify parts of the point cloud that correspond to specific buildings, we assume that the polygon defining the perimeter of the building is recovered by affine translation of the OSM position within a certain maximum radius.

Let S_i be the OSM perimeter of a building (polygon), and $T_{x,y}S_i$ be its translation by x meters

	$r_1 = 12m$		$r_2 = 8m$		$r_3 = 5.5m$	
Metric	IoU	Distance	IoU	Distance	IoU	Distance
Average ($\Delta_1 = 0.8m$)	0.791	2.411	0.774	2.269	0.764	2.577
Median ($\Delta_1 = 0.8m$)	0.779	2.232	0.801	2.128	0.773	2.235
Average ($\Delta_2 = 1.15m$)	0.791	2.266	0.784	2.197	0.759	2.806
Median ($\Delta_2 = 1.15m$)	0.787	2.065	0.799	2.056	0.783	2.876

Table 3.1: Performance of the proposed position adjustment method for different values of r: IoU and distances between centroids of estimated locations and ground truth (higher values of IoU correspond to better fit).

horizontally (east-west) and y meters vertically (north-south). Let $\hat{T}$ be the optimal translated position of the OSM building delivering maximum to our fit criterion IoU, where we consider overlap between the OSM-translated polygon and the 2d point cloud. Then the perimeter of the building in our point cloud can be found as $\hat{T}S_i$, with

$$\hat{T} = \arg\max_T \quad \textbf{IoU}(T_{x,y}S_i, D_{2d}), \quad \text{with} \quad x^2 + y^2 < r^2.$$

To establish the overlap we also take into account the density of the 2d point cloud, which is non-uniform, due to substantially higher density of points along the edges of buildings (originating from points on the vertical surfaces).

Practically we simplify the above maximisation problem by considering fixed sizes of increments for x and y translations in $T_{x,y}$. We apply two step sizes $\Delta_1 = 1.15m$ and $\Delta_2 = 0.8m$, and conduct the search of the optimal position within the distance r_i, where $i = 1, 2, 3$. We evaluate different r_i and pick 3 significant distance threshold: $r_1 = 12m$, $r_2 = 8m$ and $r_3 = 5.5m$ (see Figure. 3.4). The overall result for (r_1, Δ_1) is demonstrated in Fig. 3.5. The green polygon represents the initial OSM polygon, and the red data points are the segmented parts of the point cloud identified by our procedure.

In Fig. 3.4 we demonstrate the performance of adjustment procedure with the maximal radius increasing from $r_3 = 5.5m$ to $r_1 = 12m$ with a spatial translation step set to $\Delta_1 = 1.15m$. The initial OSM position for both highlighted buildings has a limited overlap with the point cloud data. For the Zoology building the IoU increases significantly from 0.15 (OSM after global registration) to 0.798 (after adjustment in $r_3 = 12m$). Fig. 3.4a-3.4c demonstrates the gradual improvement of the OSM position with the increase of radius. In case of reasonably isolated buildings with little vegetation around this behaviour is typical. FitzGerald buildings in Fig. 3.4d-3.4f is an example of an isolated building closely surrounded by trees, which limits the performance of the position refinement

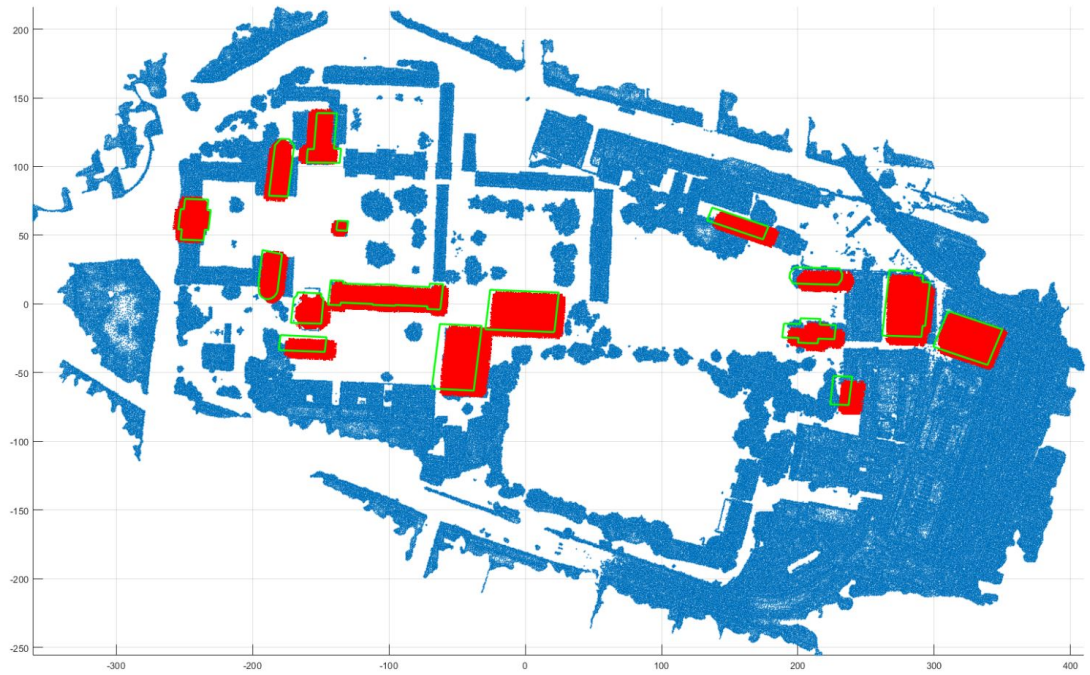


Figure 3.5: Fitting OSM to point cloud: blue patches are non-ground points from the point cloud (buildings and trees), green bounding boxes are original OSM structures, and in red are the estimated adjusted OSM positions.

procedure. Specifically, after the initial improvement from $r_3 = 5.5m$ to $r_2 = 8m$, the estimated position in $r_1 = 12m$ drops due to an adjacent cluster of dense tall vegetation which biases the estimated position.

Table 3.1 demonstrates performance of the proposed position adjustment algorithm for various values of search radius r and translation step Δ. We show the average and median for these 16 buildings and found that the IoU increases significantly. From 0.567 on average, after the global registration of OSM data, to above 0.75 after position adjustment. The distance between centroid points of polygons from 5.594 meters down to around 2-2.4 meters after automatically estimated translation adjustment $\hat{T}$. In addition, in Fig. 3.6 we demonstrate the improvement in IoU achieved with the proposed position adjustment approach for the considered individual buildings with the translation step $\Delta_2 = 1.15m$ and three radii r. The proposed greedy search strategy consistently improves the overlap between OSM shapes and point cloud data. We implement this algorithm on CPU (Intel i7 with eight threads) in MATLAB because it allows us to implement the multithreaded computations easily and automatically execute on multiple computational threads in a single MATLAB session

The segmentation process takes around 6-8 minutes depending on maximum distance r_i and step Δ_i.

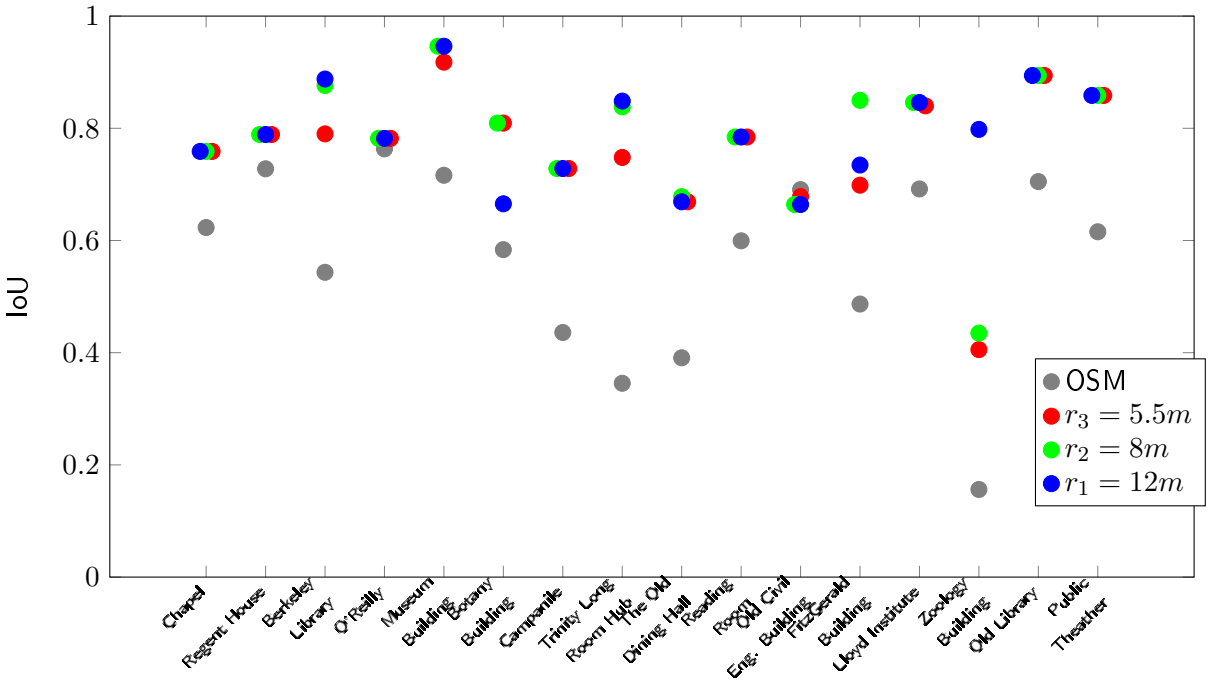


Figure 3.6: IoU per building: initial OSM (after global affine registration) and adjusted position with three radii r.

3.5 Conclusion

Our overall results show that aligning GIS (OSM) data with a 3D point cloud is promising for segmenting it in meaningful subsets of vertices. The segmentation can then be annotated for training a machine learning model. We are currently working on improving the robustness of the pipeline by integrating the colour information into the decision process.

Chapter 4

IM2Elevation

Estimating Digital Surface Model (DSM) and building heights from single-view aerial imagery is a challenging, inherently ill-posed problem that we address in this chapter using machine learning methods to overcome those issues. We propose an end-to-end trainable convolutional-deconvolutional deep neural network architecture that enables learning mapping from single aerial imagery to a DSM to analyse urban scenes. We perform a multi-sensor fusion of aerial optical and aerial LiDAR data to prepare the training data for our pipeline. The dataset quality is key to successful estimation performance. Typically, a substantial amount of misregistration artefacts are present due to geo-referencing / projection errors, sensor calibration inaccuracies and scene changes between acquisitions. To overcome these issues, we propose a registration procedure to improve LiDAR and optical data alignment that relies on Mutual Information, followed by Hough transform-based validation step to adjust misregistered image patches. We validate our building height estimation model on a high-resolution dataset captured over central Dublin, Ireland: LiDAR point cloud of 2015 and optical aerial images from 2017. These data allow us to validate the proposed registration procedure and perform 3D model reconstruction from single-view aerial imagery. We also report the state-of-the-art performance of our proposed architecture on several popular DSM estimation datasets.

4.1 Introduction

High-resolution ortho-rectified imagery acquired by aerial or satellite sensors is well known to be a rich source of information with high geolocation accuracy. These images are widely used in geographic information systems (GIS), for instance, for the detection of man-made objects (building), urban monitoring and planning. However, these optical images do not contain height information, and therefore, this limits the scope of analysis that can be done based on aerial optical capture. Point clouds, on the other hand, provide complementary 3D information. Stereo image pairs [23], structure

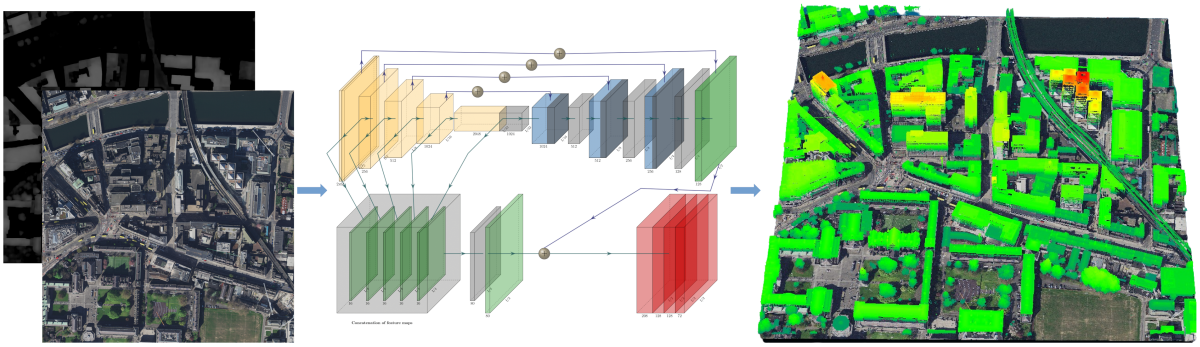
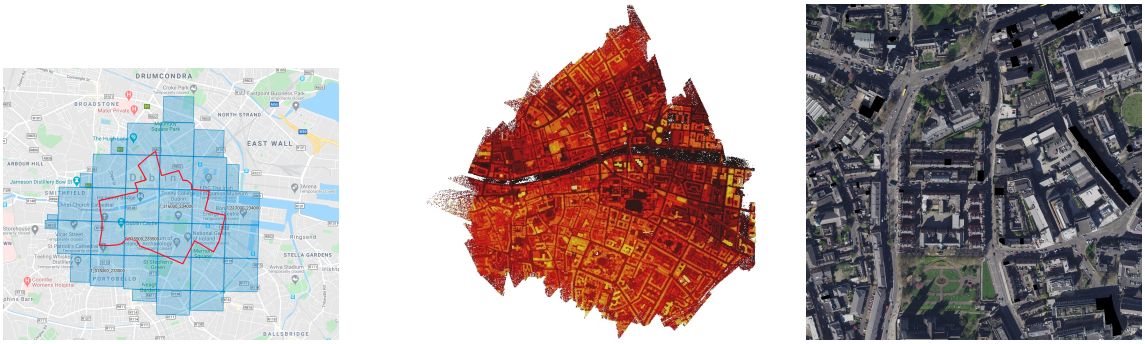


Figure 4.1: IM2ELEVATION pipeline: from single view aerial imagery to building heights / DSM. LiDAR point cloud is used solely at training phase.

from motion (SfM) [114], or Light Detection and Ranging (LiDAR) laser-scanning technology are traditionally used to obtain point clouds. These methods provide 3D information with various levels of accuracy, which can then be converted to DSM and stored as greyscale imagery. The latter can be used directly to annotate the height value on the aerial image. However, these methods require high computational resources and laborious fine-tuning. Moreover, using stereo image pairs with SfM requires image matching, which helps to estimate camera poses with different temporal intervals. The height information is extracted via triangulation from pairs of consecutive views, and therefore the single view imagery can not be used by these techniques.

In this chapter, we focus on the scenario where building height information is to be extracted in a fully automated mode from a single airborne or satellite optical image without relying on the availability of any further contemporary or historical imagery, point clouds or GIS records data. We propose a convolutional neural network (CNN) architecture IM2ELEVATION that takes a single optical image as input and produces an estimated DSM image as output, see Fig. 4.1. Our architecture is inspired by [61] and [62] and features additional skip connections and post-processing convolutional layers which deliver the highest performance on the task at hand. To enable the training process we use a set of point cloud data that serves as ground truth reference. In particular, we investigate the use of publicly available point cloud data of Dublin city centre recorded in 2015 [76] for extracting the training building height information. This is used in combination with the more recent aerial imagery collected by Ordnance Survey Ireland (OSI) in 2017. These data are of higher resolution than other alternative open source datasets, e.g. [5]. We also explore the co-registration required to perform inference on these multi-sensor data, which is needed because of the misalignment in space and time between the streams. Fig. 4.2 presents the Dublin dataset used for this study and visualises the height estimation process.

Using a supervised learning approach for converting automatically an aerial image (input) into



aerial tiles (blue) and LiDAR cover (red) 2015 LiDAR point cloud dataset [76] sample aerial image tile

Figure 4.2: Study case of building heights in Dublin (Ireland).

a DSM image (output), our first step (see Sec. 4.3) is to create a training dataset of aligned pairs (aerial-DSM). Using the resulting training set, we then train a CNN to infer building heights (see Sec. 4.5).

4.2 Related works

Using multiple sources of data for inference has a number of applications in the mapping (see Sec. 4.2.1). However, it presents challenges for machine learning techniques where the quality of the data used for training directly impacts the performance of the resulting techniques. Here we focus on buildings and the corresponding roof shapes that are important for instance for solar panels [90] and 5G deployment [2] (see Sec. 4.2.2). To capture 3D data directly is more expensive in practice than capturing multi-spectral aerial imagery therefore, some research efforts have been deployed for extracting depth information from single view images (see Sec. 4.2.3) and extended for processing aerial images to infer heights (see Sec. 4.2.4). We also report a brief review of the registration techniques applied to point clouds and aerial imagery (Sec. 4.3) since this is inevitably the first step in dealing with incoming new data, e.g. see Fig. 4.2.

4.2.1 Fusion of heterogeneous data streams

In recent years, various sources of multimedia input and GIS data have been used in numerous applications such as mapping and 3d city reconstruction. For instance, social media data and satellite imagery are used together to provide an opportunity to measure and monitor the impact of flooding [9]. Social media data has the advantage of providing up-to-date information capturing geo-located information and sentiment [27]. However, images posted on social media are too sparse and noisy for applications such as 3D city reconstruction. Street level imagery (e.g. Google Street View) has

been used for 3D reconstruction [92], for object geolocation such as poles for asset management [73] and trees for monitoring biodiversity [79]. Street level imagery, when used with building shape information extracted from OSM, also provides opportunities to generate a light 3D model of cities usable with game engine technologies [27]. Extending the point of view from street level to aerial, Liu et al. [82] proposed to use OSM to segment and propagate labels to a point cloud from drone imagery collected over Trinity College Dublin campus in 2017 [28].

4.2.2 Mapping of buildings and rooftops

Detection and segmentation of buildings in satellite and aerial imagery are standard processes for populating and efficient updating modern maps [19, 77, 88]. For instance, Lafarge, et al. [77] presented an approach for building reconstruction from a single Digital Surface Model (DSM) with buildings treated as an assemblage of simple urban structures extracted from a library of 3D parametric blocks. As an alternative to DSM, Benedek et al [19] introduced a probabilistic method which integrates building extraction with change detection in remotely sensed image pairs.

Beyond building footprint segmentation in 2D and 2.5D (DSM) aerial imagery, knowledge of roof geometry is also essential for instance, in detecting and assessing the potential of rooftop solar photovoltaic (PV) installations on local electricity networks for sustainable development. Palmer et al. [100] proposed an approach for roof PV suitability based on slope and aspect using aircraft-based LiDAR data, building footprint data, GIS tools, and aerial photography. Song et al. [121] proposed to extract the 2D rooftop outlines and 3D rooftop parameters retrieved from high-resolution remote sensing image data and DSM considering five rooftop categories (flat rooftops, shed rooftops, hipped rooftops, gable rooftops and mansard rooftops).

4.2.3 Monocular depth estimation from images

While stereo vision has been intensely researched to use triangulation cues from two or multiple images, it can not be applied to the case where only a single image is available. Instead, there are numerous monocular cues, such as texture variations and pixel's gradient, occlusion, and colour/haze, which can be used to infer the underlying 3D structure. Single-view depth estimation can be classified into two major groups, specifically random field-based approaches and deep convolutional neural networks (DCNNs) approaches. Saxena et al. [113, 112] segmented an image into small patches (super-pixel), and Markov Random Fields were applied to infer depth for each of the segmented patches. Their model is trained based on several features at different scales [113, 112]. A similar approach by Wei et al. [143] introduced a hierarchical representation of the scenes with a depth prediction model based on Conditional Random Fields (CRF). The latter encoded interactions within

and across different layers, jointly exploiting local and global 3D information. More recently DCNNs-based models have been actively investigated [40, 84, 134, 85, 78, 89]. Eigen et al. [41] proposed to use two pipelines of DCNNs to extract features at different scale levels: one that regressed global depth structure from a single image and another for capturing the high-level features. Both outputs were concatenated and are collectively refined in the final layer. This has later been extended to handle multiple tasks like semantic segmentation and surface normal estimation [40].

Laina et al. [78] and Xu et al. [134] demonstrated the use of CRFs with DCNNs-based models. These hybrid DCNNs methods capitalise on exploring the strength of pairwise pixel interaction, whereas the DCNNs only consider the unary potentials during training. In a similar fashion, Long et al. [85] leveraged connected CRF as part of the full convolution network layer to jointly maximise the posterior on predicting semantic labels. The encoder-decoder framework has been a common trend in DCNN architecture [70, 57, 63, 61, 89]. The encoder transforms the imagery into a latent space which contains high-level features of the imagery. The decoder is to increase the spatial resolution of the feature maps generated by the encoder. This enables the networks to regress the input to the desired output. Laina et al. [78] used the ResNet-50 [57] as the DCNN architecture's backbone and replaced the fully connected layer to the up-projection blocks which act like the reverse operation of pooling, scaling up the spatial resolution to half the size of the input. Mal et al. [89] used up-projection blocks [78] in deconvolutional layers to learn the mapping from a sparse depth map to a dense depth map. Both [89] and [78] used similar encoder-decoder fashion architectures to predict depth information from a single image; however, their results are underperforming in terms of preserving the object shapes.

Hu et al. [62] combined the up-projection blocks with the concept of CNN skip connections [39]. In addition, the features from the encoder are fused into the same size block as a multi-feature fusion block (MFF). The MFF is then concatenated to the output of the decoder. Furthermore, they use gradients and normals as part of the loss function so that the designed network is more aware of the edges and shapes during training.

4.2.4 Aerial image height estimation

Unlike monocular depth estimation in computer vision, there has been relatively little on estimating depth from a single aerial image. The methods that are used in monocular depth prediction can be considered as the problem of estimating the distance between the sensor to the observed scene in remote sensing. Height estimation from a single view is an ill-posed problem: a single 2D image may have an infinite number of possible 3D scenes. The ambiguity lies in mapping from 3 channel RGB into a channel height value. Fully convolutional-deconvolutional network architectures are a useful

tool that is capable of guiding the model through the process of learning this ambiguous mapping with considerable accuracy. Alidoost et al. [12] proposed to regress RGB image to DSM, together with linear line structure of the roof in 3 different classes: eave, ridge, and hip lines. The lines were used as additional information in their building reconstruction process.

Architectures with skip connection are capable of keeping the global edge features and have been recently used for buildings height estimation [93, 13]. Similar architectures were used to jointly estimate height and semantic labels [123, 30] in order to improve accuracy by using the training information for these two complementary tasks. While the skip connection has been widely used to directly concatenate the same spatial resolution features from encoder to decoder, we establish the superior performance of multi-scale feature fusion from encoder blocks (see Fig. 4.7).

Ghamisi et al. [46] exploited conditional Generative Adversarial Network (cGAN) to generate a synthesised DSM from a single aerial image. Bittner et al. [21] used cGAN to refine the building roof surface by regressing DSM from stereo aerial image to level of detail (LoD) 2 building shape.

4.3 Data fusion

We outline the employed data pre-processing as follows: in Sec. 4.3.1 for LiDAR data, and in Sec. 4.3.2 for aerial imagery. The registration procedure we propose to align our two heterogeneous and asynchronous datasets (aerial images 2017 and LiDAR point cloud 2015) is presented in Sec. 4.4, and then validated to improve robustness (see Sec. 4.4.1). The pre-processing allows us to create a dataset for training and testing our CNN model that converts aerial images (input) into DSM (output).

4.3.1 Preprocessing of LiDAR data

Our purpose here is to generate DSM ground truth from the available LiDAR point clouds. These DSM images are then used as the reference for training and testing our CNN.

In our case study, we work on a 2015-captured LiDAR dataset that covers the city centre of Dublin, with around 5.6 km^2 including partially covered areas [76]. The flight altitude was approx. Three hundred meters, and there are 41 flight path strips in total. The final LiDAR point cloud was generated from the registration of points from these strips. As this LiDAR point cloud contains more than 1.4 billion points, it is split into 0.25 km^2 tiles to be loaded and processed efficiently. The point cloud density inside the projected area varies from 238 to 348 $points/m^2$. There is around 3.5 km^2 complete point cloud coverage equivalent to 14 tiles. The average accuracy of this point cloud is 0.03 meters with a maximum of 0.3 meters deviation. Tiles are back-projected to DSM (output of our neural network). The ground control sample distance (GSD) of the generated DSM is scaled to 15 ground resolution which is the resolution of the aerial images employed. The generated DSM from

LiDAR point cloud has holes caused by the low density of points in certain areas. To enhance the DSM completeness, we fill those holes by interpolating from the closest available points. In addition, the DSM is saved as a 16-bit image format which preserves more details about the local gradient (see in Fig. 4.3).

The return-pulse intensity is also obtained from the LiDAR point cloud. It is merged with DSM (Fig. 4.4) as an additional source of information for validating our registration process. Our unit voxel volume is set to $15cm \times 15cm \times 15cm$ to be at the same resolution as the aerial imagery. The density of the points in the LiDAR point cloud is observed to be higher on vertical walls that translate as edges in aerial imagery and DSMs.

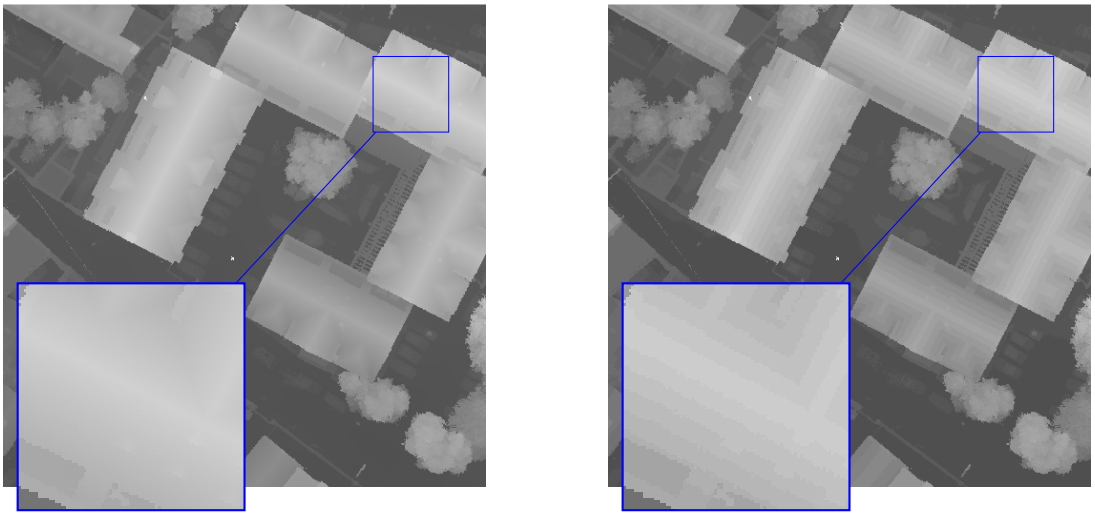


Figure 4.3: The roof has a smoother gradient in a 16-bit image (left) compared to the 8-bit one (right). 8-bit DSM can only capture details up to 1 m resolution, whereas 16-bit DSM is capable of capturing at 0.15 m resolution.

4.3.2 Preprocessing of aerial ortho-rectified imagery

The aerial image we use in our study is captured by OSI in 2017 and covers the entire extent of the LiDAR 2015 dataset at 15 cm geometric resolution. Note that the LiDAR dataset 2015 [76] also provides aerial imagery; however, it only partially covers Dublin city (see Fig. 4.2). Therefore, we deal with two distinct datasets, and there are two major challenges: (1) The change of scenes between the two captures (in 2015 and in 2017), and (2) the two data streams are not properly aligned with each other even though both of them are geo-referenced. Small deviations can be observed that originate from different projection methods and sensor calibration errors.

We first remove the areas that change substantially between the acquisition dates of the two datasets, this mainly occurs on dynamic objects, e.g. cars and buses. Only areas with DSM values

above 2.5 meters from the ground are kept to mitigate such discrepancies.

We then convert RGB optical images to greyscale so that they can be used to perform registration with DSM via Mutual information registration (explained Sec. 4.4). The return-pulse intensity and DSM images are merged (see Fig.4.4) and then used with the optical image for validation of our MI registration quality (explained Sec. 4.4.1). This approach is chosen because DSM has a more similar distribution of intensities to optical imagery than return-pulse intensity, which leads to a better performance in MI registration. Moreover, fusing the DSM and the return-pulse intensity retains more distinctive features alongside edges which boosts the Hough line extractor's performance. The return pulse is generated by the reflected pulse from the object's surface. One emitted laser pulse can return to the lidar sensor as one or many returns.

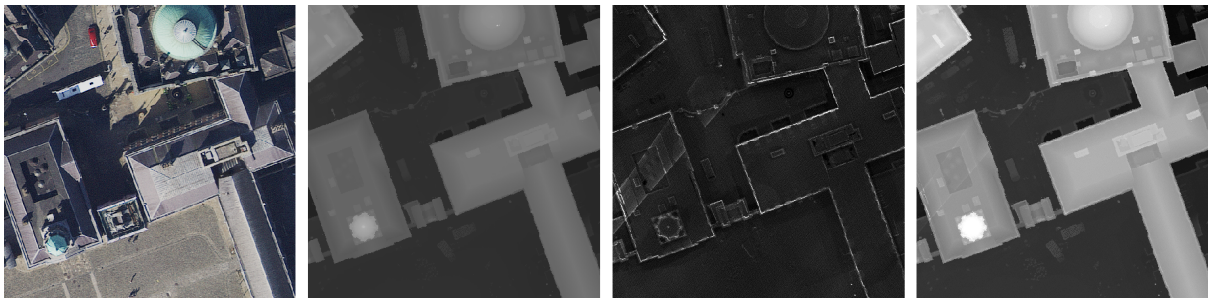


Figure 4.4: From left to right: optical imagery, DSM, return-pulse intensity and, Fusion of DSM and return-pulse intensity. The optical image and the DSM are used in the MI registration process, whereas the fusion and the optical image are used in Hough validation. Both DSM and return-pulse intensity are derived from the point cloud source

4.4 Registration with Mutual Information

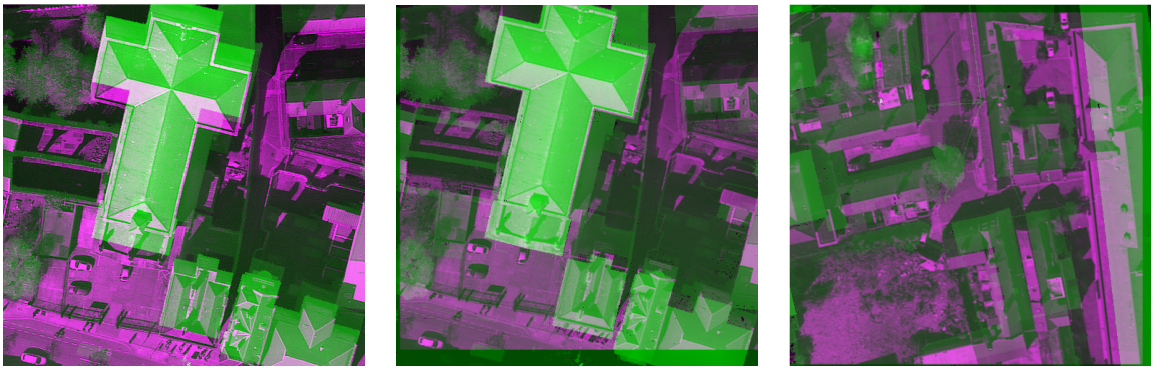
Mutual information is employed to carry out registration between aerial imagery and DSM. MI [101] measures the statistical similarity between datasets. The matching similarity is based on the intensity of distribution within images. The definition is as follows:

$$MI(I^P, I^D) = H(I^P) + H(I^D) - H(I^P, I^D) \tag{4.1}$$

$$H(I^P) = -\sum_{i=1}^{N} p_1^i \cdot \log(p_1^i) \ , \ H(I^D) = -\sum_{i=1}^{N} p_2^i \cdot \log(p_2^i) \tag{4.2}$$

$$H(I^P, I^D) = -\sum p(i,j) \cdot \log(p(i,j)) \tag{4.3}$$

where $H(I)$ is the marginal (Shannon) entropy of image I [116], N is the number of bins used for the histograms of the images (estimated distribution p), and $H(I^P, I^D)$ is the joint entropy.



Initial misalignment | Successful registration | Failed registration

Figure 4.5: MI registration between return-pulse intensities (green) and optical image (purple) may fail when the scene content is significantly different between the two data sets.

The aerial image I^P is chosen as the reference source on which the DSM image I^D is aligned. The statistical dependence between I^P and I^D is computed using a probability density function (pdf) estimated via histogramming using a non-parametric approach. One-plus-one evolutionary algorithm [125] was used in the optimizer. It perturbs the matrix parameters in different directions to find the maximum MI value. In the case of misregistration, the output from the joint pdf $p(i, j)$ yields a large negative number, which increases the value of the joint entropy. After appropriate registration, the images are expected to yield the smallest joint entropy. A parametric registration transformation between the images is formulated as a translation operation. Disabling rotation and scaling operations is a trade-off to enable efficient and robust registration, as otherwise, the process is highly affected by smaller objects other than buildings. The translation T is estimated as follows:

$$\hat{T} = \arg\max_{T} MI(I^P; I^D) = \arg\min H(I^P; I^D) \ , \ \hat{T} \in \mathbb{R}^2. \tag{4.4}$$

We ran 1000 iterations to get the optimal solution $\hat{T}$. GPS location available with the images is used to initialise T. Note that the initial level of registration between the available LiDAR and optical imagery, both of which are geo-referenced, allows for only approximate matching and we resort to an iterative procedure to improve the quality of local matching. We denote the aligned DSM image as $\hat{I}^D$.

The registration process can fail, as demonstrated in Fig. 4.5 if a scene has undergone substantial changes in between the acquisitions. To detect such situations a Hough line extractor is applied to aerial images and the corresponding fusion of DSM and intensities (Sec. 4.4.1). We compute the distances between the Hough detected lines in the two modalities and validate matching pairs. Invalid examples were adjusted via their neighbours' mean vector. This is used as the objective function to ensure the quality of the registration samples. The aim is to create a good training dataset to train

our network: the quality of the training data is crucial for the high performance of the resulting neural network.

4.4.1 Patch adjustment via Hough lines validation

To enhance the quality of pairs $\{(I_k^P, \hat{I}_k^D)\}$ for training our deep neural network, we exploit the Hough space to retain the good pairs of images registered with MI. Given two images I^P (optical imagery) and I^Q (fusion of DSM and return-pulse intensity), two sets of Hough lines, noted $V^P = \{\rho_i^P, \theta_i^P\}_{i=1}^{N^P}$ and $V^Q = \{\rho_i^Q, \theta_i^Q\}_{i=1}^{N^Q}$ are computed (see on Fig. 4.6). The distance between parallel lines ($\theta^Q = \theta^P$) between I^P and I^Q is computed as:

$$e = \|\rho^P - \rho^Q\| \tag{4.5}$$

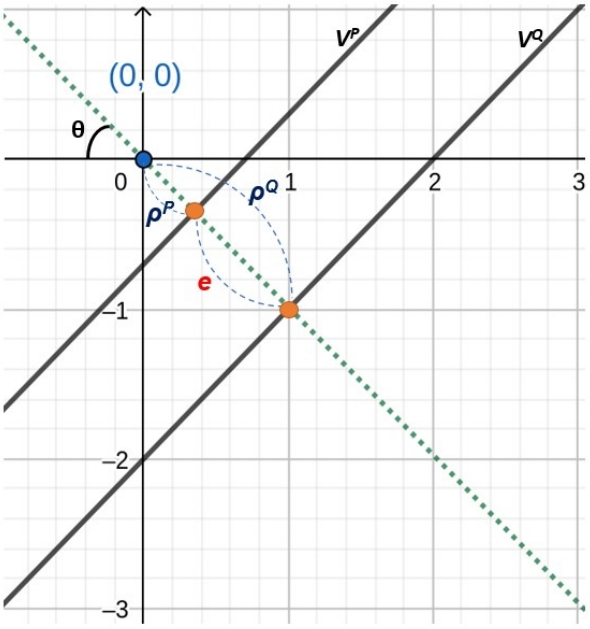


Figure 4.6: An example of parallel Hough lines' associated parameters and the distance e between V^P and V^Q. In practice, V^P is a fixed-line while V^Q move in accordance with MI registration.

Denoting e and e' the distance between the two lines before and after registration, respectively, we validate a pair of patches having n corresponding lines when

$$\sum_{i=1}^{n}(e_i - e_i^{'}) \begin{cases} > 0, & \text{valid-registered pairs} \\ \leq 0, & \text{not valid-registered pairs} \end{cases}$$

Parallel lines were grouped together by searching the closest matching lines within a range $e < \delta$ before registration to reduce the impact of failed registration. To correct the detected cases of failed patch registration, the missing translation vectors are interpolated between the adjacent correctly registered neighbour patches (this is referred to as adjustment in the following).

4.5 CNN Network design

We propose a deep learning architecture based on that presented in Hu et al. [62]. Specifically, we use a fully convolutional-deconvolutional network presented in Fig. 4.7 that regresses an RGB (3 channel) image to a DSM image (1 channel). A Squeeze-and-Excitation Network [61] is used as an encoder to extract features, generating five encoding blocks.

The encoder (convolutional layers) $E0 - E4$ extracts feature with the resolution of 1/2, 1/4, 1/8, and 1/32 of the original image. The decoder (deconvolutional layers) $D0 - D3$ up-samples the scale feature from 1/32 to 1/2 of the original image size. We use the decoder proposed by Laina et al. [78] with up-sampled feature maps concatenated with the corresponding features from the encoder, followed by 1×1 convolution layers $D0\_1, D1\_1, D2\_1$, to reduce the number of channels. The latter blocks mix the weights and reduce the number of channels. Multi-scale features $M1 - M5$ are extracted from an encoder and re-sampled to the constant scale of 1/4 of the initial image resolution, then mixed via 1×1 layer $F0$ and up-sampled to the scale of 1/2 of the input in $F1$. For visualisation purposes, examples of several multi-scale feature maps are presented in Fig. 4.8.

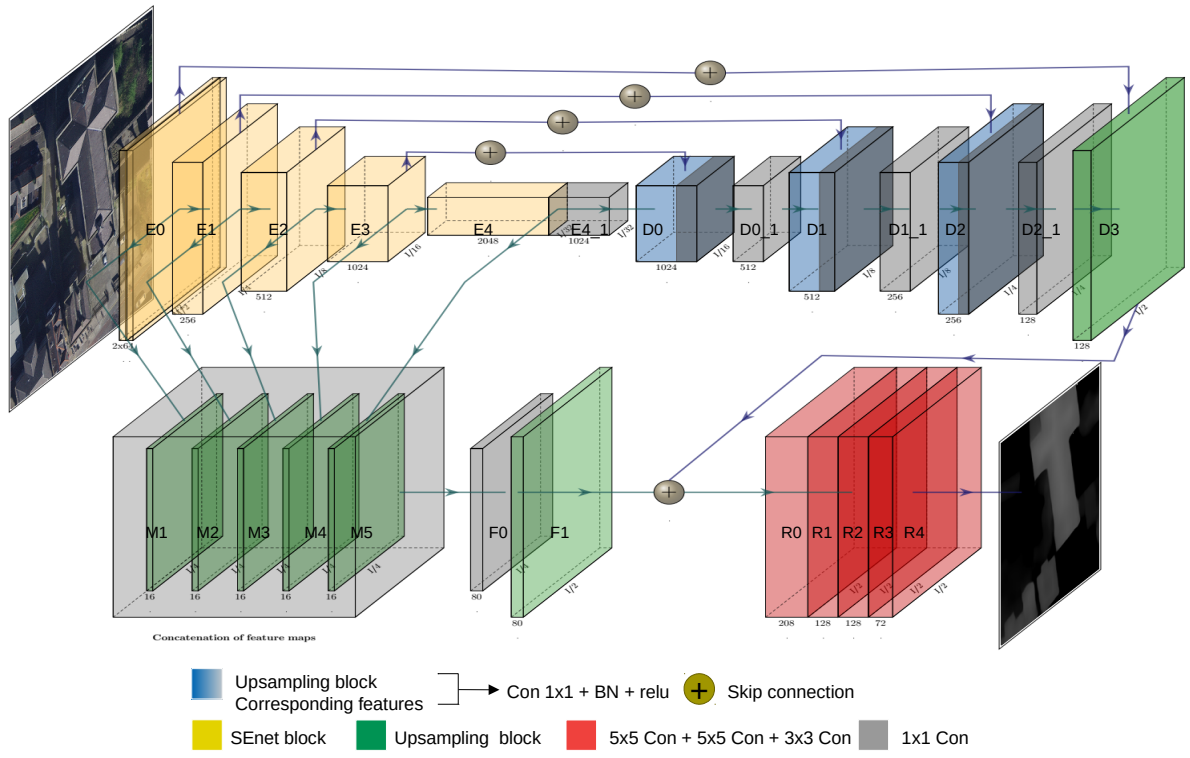


Figure 4.7: IM2ELEVATION architecture with extra skip connections and post-processing layers. E: down-sampling block. D: up-sampling block. M: multi-fusion block. F: multi-fusion block processing. R: convolution layers.

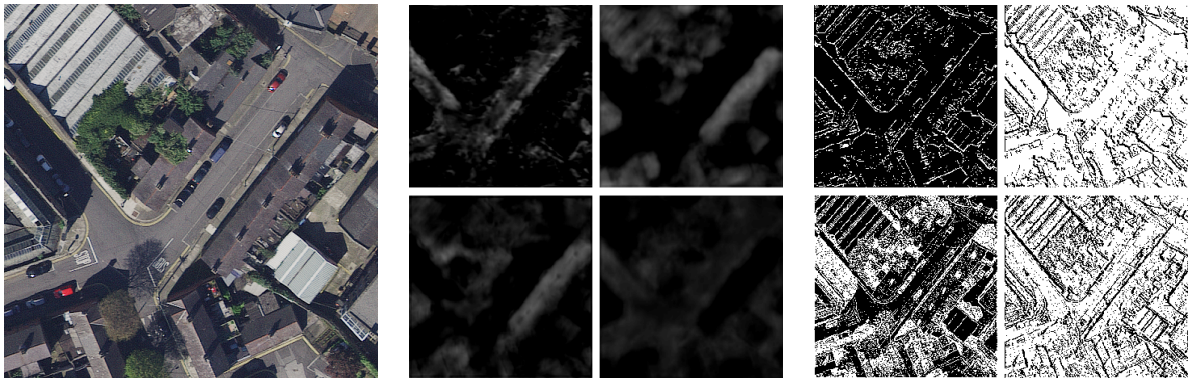


Figure 4.8: From left to right: aerial image, CNN features from Multi-feature block, and CNN features from the first layer of encoding block.

Finally, the last layer of deconvolution $D3$ is concatenated with $F1$, thus forming $R0$, and fed into convolution layers $R1 - R4$ in order to refine the final prediction map. Each convolutional except $R4$ layer features batch normalisation and rectified linear unit (ReLU) activation function. The details of the input/output channels and the feature size are reported in Table 4.1. Further details on standard architectural elements can be found in [62, 78].

Our architecture employs a skip connection that concatenates features directly from the encoder block with those from the multi-scale features block. We observe a significant empirical improvement in our results due to this strategy. See Table 4.3.

The network loss function guides the training process to obtain the best fitting function modelling the training data. We construct the loss that consists of L_1 norm (Eq. 4.7), surface normal (Eq. 4.9) and spatial gradient (Eq. 4.8) as follows:

$$L = l_{depth} + \lambda\, l_{grad} + \mu\, l_{normal} \tag{4.6}$$

where λ, $\mu \in \mathbb{R}^+$ are weight coefficients. Here we have:

$$l_{depth} = \frac{1}{n}\sum_{n=1}^{i} F(e_i)\,, \quad e_i = \|\hat{h}_i - h_i\|_1, \quad F(x) = \ln(x+\alpha), \tag{4.7}$$

e_i is the L_1 norm between the estimated height $\hat{h}_i$ and ground truth height h_i. $\alpha > 0$ is a slack parameter to ensure $F(e_i) \in \mathbb{R}$ has a lower bound (in practice we always set $\alpha = 0.5$). In order to make the network more aware of the edge structure, we introduce l_{grad} defined as:

$$l_{grad} = \frac{1}{n}\sum_{n=1}^{i}\left(F\left(\frac{\partial e_i}{\partial x}\right) + F\left(\frac{\partial e_i}{\partial y}\right)\right) \tag{4.8}$$

where (x, y) are the spatial coordinates in the residual image e.

Layer	Output size	Input/C	Output/C
E0	440X440	3	64
E0_1	220X220	64	128
E1	110X110	128	256
E2	55X55	256	512
E3	28X28	512	1024
E4	14X14	1024	2048
E4_1	14X14	2048	1024
M0	110X110	64	16
M1	110X110	256	16
M2	110X110	512	16
M3	110X110	1024	16
M4	110X110	2048	16
F0	110X110	80	80
F1	220X220	80	80
D0	28X28	2048	1024
D0_1	28X28	1024	512
D1	55X55	1024	512
D1_1	55X55	512	256
D2	110X110	512	256
D2_1	110X110	256	128
D3	220X220	128	208
R1	220X220	208	128
R2	220X220	128	128
R3	220X220	128	128
R4	220X220	72	1

Table 4.1: IM2ELEVATION size of feature maps, and input/output channel based on SENet[61]

l_{normal} is a cost on the normals to the surface of the estimated DSM with respect to its ground truth:

$$l_{normal} = \frac{1}{n}\sum_{n=1}^{i}\left(1 - \frac{\langle n_i^d, n_i^g\rangle}{\sqrt{\langle n_i^d, n_i^d\rangle}\sqrt{\langle n_i^g, n_i^g\rangle}}\right) \tag{4.9}$$

$\langle\cdot,\cdot\rangle$ denotes the inner product of vectors, which infers the orientation of the surface. This cost penalises the subtle changes of gradient which may not be captured in Eq. 4.8. Besides, this guides

the network to learn more geometric features.

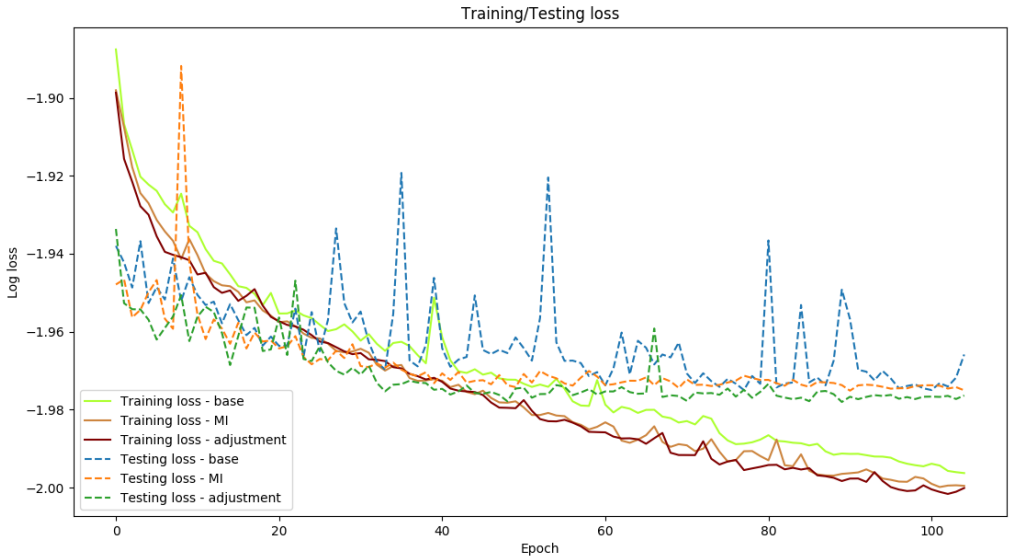


Figure 4.9: The training and testing loss on original data (base), MI-registered, and after Hough line patch adjustment. The registered data demonstrate more stable convergence than non-registered data.

4.6 Results

Our dataset consists of 14 tiles, and each tile is split into 169 patches. Each patch is 500×500 pixels in size with 250 pixels overlapping for the purpose of increasing training data volume. Of the original 2366 patches, 1999 patches are used for training, and 367 patches are used for testing and for comparisons between several pre-processing pipeline scenarios: no registration (base), registration (MI) and registration with invalid patch adjustment (adjustment). In addition, we augment the training set by randomly flipping and jittering the training data. Adam solver is used with a weight decay of 0.001 and a learning rate of 0.0001. The learning rate is dropped by 10 per cent in every five epochs. The CNN network is done with PyTorch package[1]. Fig. 4.9 demonstrates the evolution of the training/test loss throughout the training process: both MI and patch-adjusted datasets show a steady loss decrease, whereas the base dataset (without pre-processing) has large fluctuations (unstable) training loss. Similar behaviour has been observed on the training/test Mean Absolute Error (MAE). We notice that beyond epoch 60, the test performance plateaus.

In Sec. 4.6.1 we assess the registration with Mutual Information. Sec. 4.6.2 presents our CNN

[1]The implementation will be made available upon publication at `https://github.com/speed8928/IMELE`.

improved performance when using registered, and non-registered training sets.

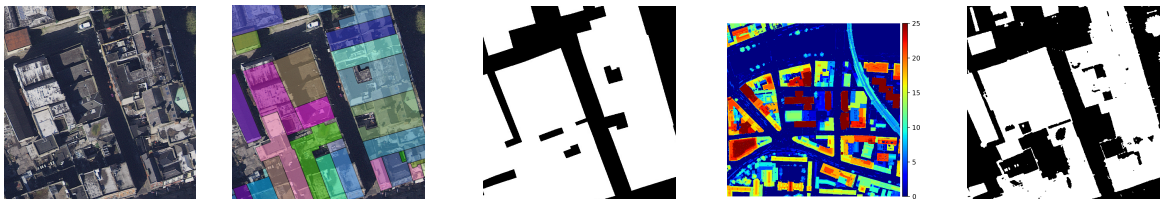
4.6.1 MI registration and Hough line validation

To evaluate the accuracy of registration, the vector layers for buildings provided by OSI are used as ground truth after being transformed into binary maps. This additional data source is used solely for the purpose of validating the performance of the registration pipeline and is not part of CNN training. For evaluating the performance of MI and the effect of validation with Hough transform, 11 tiles were selected. For each tile of an image, 169 patches were generated, giving the total number of testing samples $K = 11 \times 169 = 1859$.

Fig. 4.10 illustrates the data used in our experiments. For each test sample $k \in \{1, \cdots, K\}$, see Fig. 4.10(a), the total number of valid pixels associated with ground truth building label is denoted as n_g^k. The DSM is converted to a binary map for buildings by setting all pixel values to 0 if these have values inferior to 2.5 meters above ground and set to 1 otherwise (if superior to 2.5 meters). The intersection defined by Eq. 4.10 gives the average of the proportion of pixels from DSM building binary map (denoted M_h^k, see Fig. 4.10 (e)) corresponding with the ground truth building binary map (denoted M_g^k, see Fig. 4.10 (c)), normalized by n_g^k (the valid number of pixels from M_g^k):

$$\text{Intersection} = \frac{1}{K} \sum_{k=1}^{K} \frac{M_h^k \otimes M_g^k}{n_g^k} \tag{4.10}$$

The higher value of intersection indicates a better registration between the two modalities of data. The higher intersection value indicates a better registration between the two modalities of data. To maximize the intersection, we translate DSM map (M_h^k) multiple times and then perform pixel-wise multiplication with the ground truth map (M_g^k).



(a) Sample k (b) building shape (c) ground truth M_g^k (d) DSM (e) binary DSM M_h^k

Figure 4.10: Example of data used for pre-processing and training.

Table 4.2 highlights the quantitative improvements associated with the use of MI and Hough validation pre-processing steps. Misregistered patches identified by Hough transform are adjusted by a local translation that allows more meaningful computation of intersection for this evaluation (Method III). The value of MI after patch adjustment declines a little, whereas the value of Intersection

increases. Further validation of pre-processing is reported in the next Section in conjunction with the proposed CNN.

Methods	MI registration	data adjustment (Hough validation)	MI↑	Intersection↑
I	✗	✗	0.7650	0.8912
II	✓	✗	**0.9235**	0.9880
III	✓	✓	0.9193	**0.9926**

Table 4.2: MI and Intersection obtained by applying registration. $n_g = 1.34$bn. # patches=1859. Note that we employ only a subset of training data due to the partial coverage of the building binary map.

4.6.2 Height inference with CNN

We now report the performance of the height inference CNN on the pre-processed Dublin dataset. The Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) are presented in Table 4.3. The comparison with the baseline architecture Hu et al. [62] demonstrates that the proposed CNN improvements are of significance for the height estimation problem. We note that using registered data for training only improves the RMSE but deteriorates the MAE in comparison to non-registered training data. However, an improvement in both can be found in registered data with patch adjustment.

Method	Preprocessing	GSD	MAE(m)↓	RMSE(m)↓
Hu et al. [62]	none (I)	15cm/pixel	1.99	5.04
	MI registration(II)		2.08	4.12
	MI, patch adjustment (III)		1.93	3.96
Hu et al. [62] + skip connection	MI, patch adjustment (III)	15cm/pixel	2.40	4.59
IM2ELEVATION	MI, patch adjustment (III)		**1.46**	**3.05**

Table 4.3: DSM estimation: Hu et al. [62] vs. IM2ELEVATION (proposed) with different types of registration between LiDAR and aerial data. The loss employs $\lambda = 1, \mu = 1$ for all methods.

We tested our network on popular remote sensing datasets, including IEEE GRSS Data Fusion Contest dataset 2018 (DFC2018) [1, 135] and ISPRS Potsdam and Vaihingen datasets [3, 4], see Table 4.4. After retraining on these data, our network outperforms other state-of-the-art methods [30, 13, 12, 46] on DFC2018 and Potsdam datasets. We do not apply any pre-processing developed throughout Sec. 4.3 to these data to ensure comparability with benchmark methods. We

Methods	Training input	GSD	MAE(m)↓	RMSE(m)↓
IEEE DFC2018 dataset				
Carvalho et al. [30]	DSM	5cm/pixel	1.47	3.05
Carvalho et al. [30]	*DSM + semantic*		*1.26*	*2.60*
IM2ELEVATION, $\lambda = 1, \mu = 1$	DSM		**1.19**	2.88
ISPRS Vaihingen dataset				
Amirkolaee et al. [13] with ZCA whitening*	DSM	8cm/pixel	-	2.87
IM2ELEVATION, $\lambda = 0, \mu = 0$	DSM		2.96	4.66
ISPRS Potsdam dataset (nDSM)				
Amirkolaee et al. [13] with ZCA whitening*	DSM	8cm/pixel	-	3.46
Alidoost et al. [12]	DSM		-	3.57
Ghamisi et al. [46]	DSM		-	3.89
IM2ELEVATION, $\lambda = 0, \mu = 0$	DSM		1.52	2.64

Table 4.4: State-of-the-art comparisons on popular datasets. *[13] employs ZCA whitening on training data, so the results may not be compared directly. *[30]* performs multitask learning.

DSM method	GSD	MAE(m)↓	RMSE(m)↓
OSI stereo	15cm/pixel	2.36	4.90
IM2ELEVATION (with MI, patch adjustment)	15cm/pixel	**1.46**	**3.05**

Table 4.5: Monocular DSM estimation outperforms the stereo imagery-based method

employ approx. 25% randomly selected tiles for testing, and the rest is used to train our pipeline for each of the three considered benchmark datasets. Note that in our work, we employ the DSM information solely and make no use of semantic labels, which is an additional and expensive (effort-wise) source of complementary information. Nevertheless, our method still outperforms some of the other benchmark methods, which perform multitask learning. In particular, IM2ELEVATION outperforms other methods in MAE on the DFC2018 dataset. Among the single-task learning models, our model reaches the highest accuracy in $RMSE$ in Potsdam dataset. We also observe that the accuracy obtained with the Dublin dataset is in general at a comparable level which suggests that this dataset is reasonably balanced and our results can be generalised.

We demonstrate the obtained DSM estimation results of a full tile in Fig. 4.11. It can be readily observed that the building contours present high variance in general. This can partially be attributed to remaining artefacts after registration and ortho-rectification processes. Buildings substantially taller than average in the scene appear to be more challenging for monocular height estimation as well.

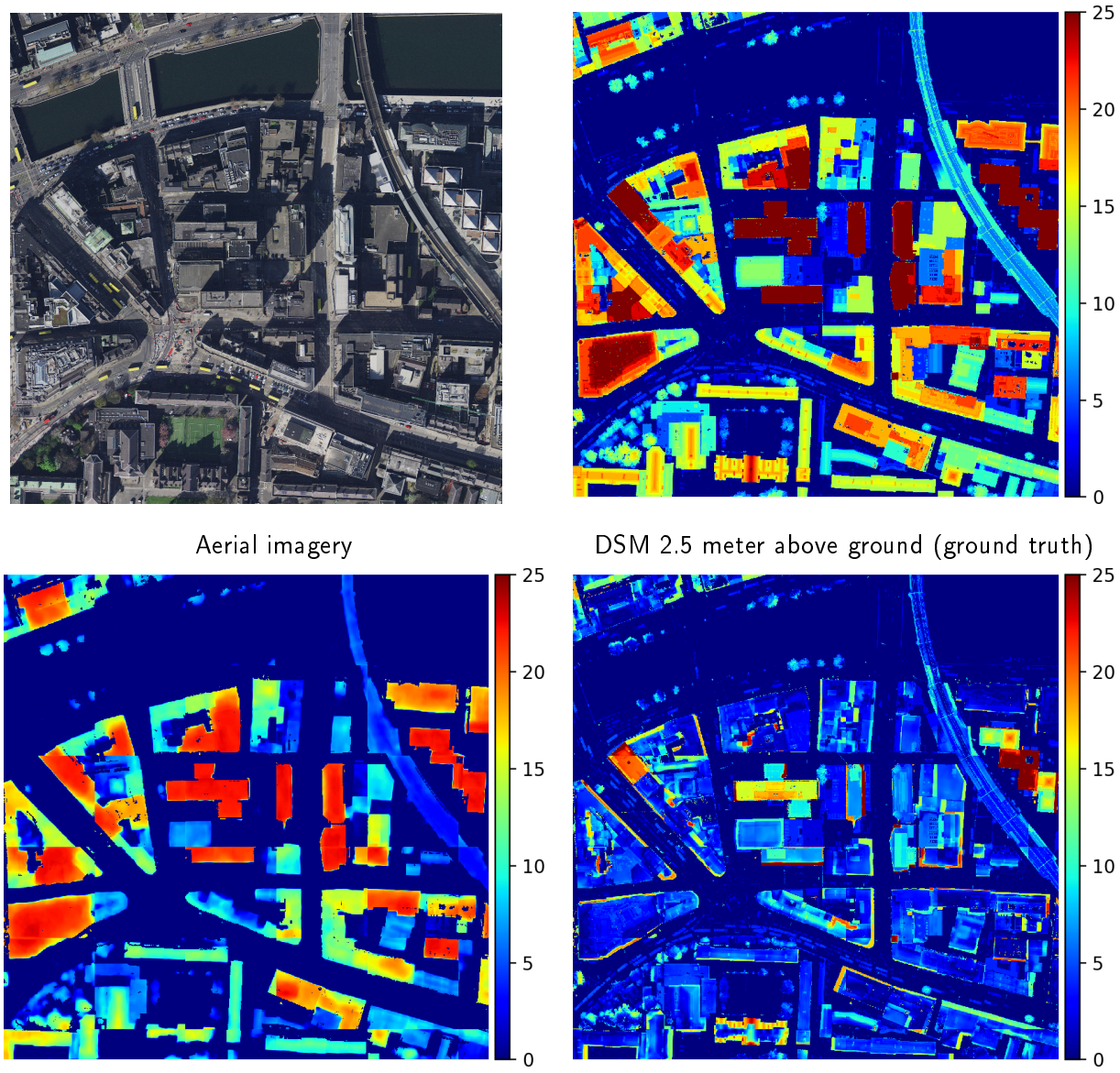


Figure 4.11: DSM of a 500x500 meters tile in the Dublin inner city centre with northing of 316000 to 316500, and easting from 234000 to 234500, in TM65 projection. As we can see from MAE (the ground truth subtracts prediction DSM), large differences happen between the edge of the building and the high buildings.

4.6.3 Height inference from stereoscopic imagery

The height inference on stereo images is reported in this section to compare with our height inference performed on the monocular set of airborne imagery. In remote sensing, structure from motion (SFM) is often adapted to infer the elevation of the landscape [42]. The point cloud can be generated by rectifying a selected pair of images, followed by computing the disparity map which is similar to DSM but with an unknown scale. The composite point cloud data generated by historical stereo images are fused with the 2D map to normalise the scale. The same resolution of imagery with 15 cm/pixel GSD is used. The density of the generated point cloud is 40 $points/m^2$. The point cloud is back-projected to 2D imagery as DSM, with a scale GSD of 15cm/pixel. The DSM from stereo images is evaluated by using the testing dataset; the outcome is reported in Table. 4.5.

4.7 Discussion

We present a panel of detailed examples of DSM reconstruction obtained with our pipeline and from stereo imagery in Fig. 4.12. The heat maps highlight the difference between our estimated DSM and the corresponding ground truth DSM. We observe that our estimates present a less salt-and-pepper type of noise and succeed in extracting the outlines of the buildings on a par with the stereoscopic imagery input.

To investigate the potential of height reconstruction for visualisation purposes, we now report the results of mesh reconstruction in Fig. 4.13. Specifically, the estimated DSM is back-projected to a 3D point cloud using pixel geo-referencing information and elevation value, and reconstructed to a 3D mesh by 2.5D Delaunay triangulation [43]. The points are triangulated in 2D space by using their row and column geolocation coordinates. The edges between vertices are inherited from pixels. Finally, the input aerial images are overlaid with the resulting building mesh. The Eye-Dome Lighting shader [24] is applied to facilitate depth visualisation.

In Fig. 4.14 we attempt to perform DSM estimation on a different image dataset. To this end, we collect Google Maps satellite images covering the same geographic extent as the crops in Fig. 4.12 using Google API. We then perform inference on these without any additional retraining or fine-tuning of our CNN pipeline. We observe that the estimation results are substantially worse due to the different properties of the Google images as the result of a completely different (and not explicitly disclosed) pre-processing applied to these images. This highlights the necessity for pipeline fine-tuning when the image source changes. As demonstrated by the results reported in Sec. 4.6.2 in Potsdam and DFC2018 experiments, once retrained IM2ELEVATION pipeline is capable of producing state-of-the-art results.

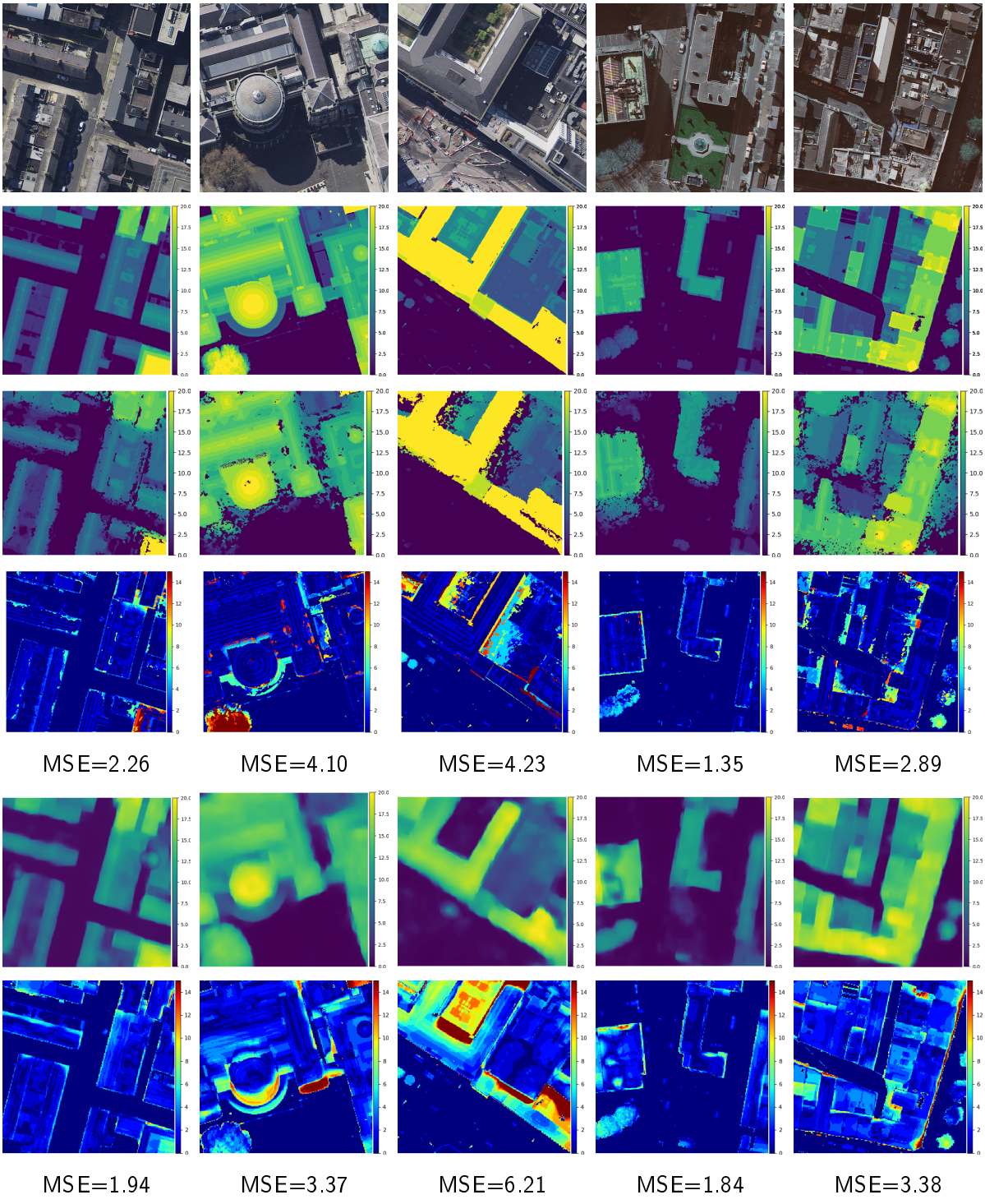


Figure 4.12: DSM from stereo imagery (source OSI) vs DSM from IM2ELEVATION. From top to bottom (rows 1 to 6): (1) aerial image input; (2) ground truth DSM; (3) DSM generated from stereo images; (4) heat map of the difference between (2) and (3); (5) DSM from IM2ELEVATION; (6) heat map of the difference between (2) and (5). The Colour scale is in meters. Note the strong impact of a tree canopy in the second column sample with IM2ELEVATION DSM reconstructing this element better.

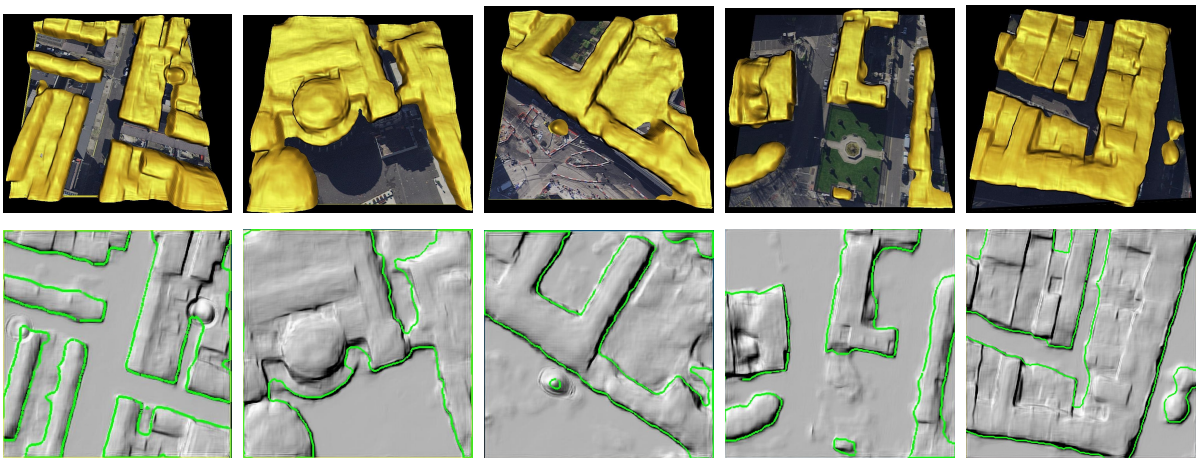


Figure 4.13: Further processing of our DSM estimates for mapping:: 3D reconstruction and 2D building footprint detection. 3D mesh is obtained via Delaunay triangulation with shading.

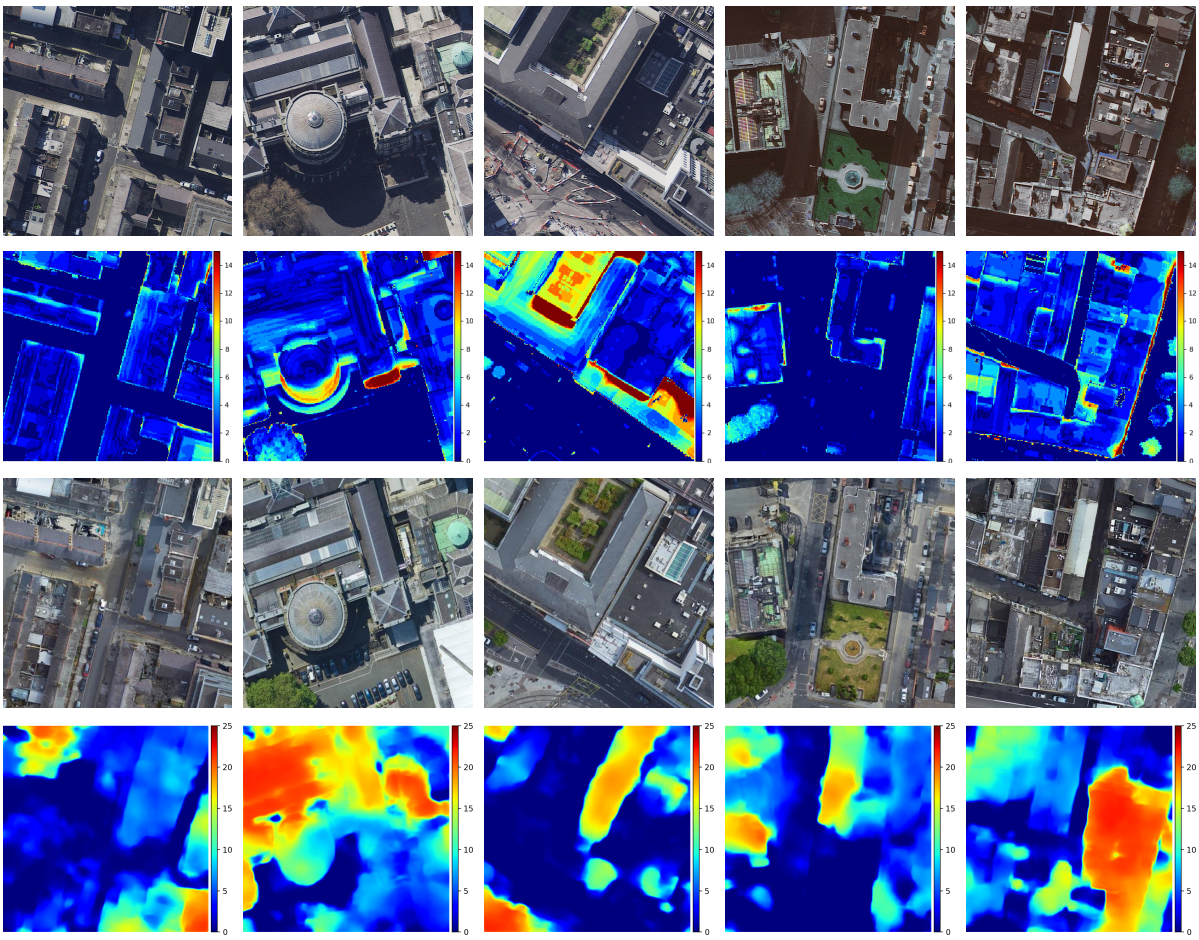


Figure 4.14: Comparison of DSM reconstruction (without retraining) from OSI aerial imagery (row 1) and Google Maps satellite images (row 3) and the corresponding error of estimations heat maps.

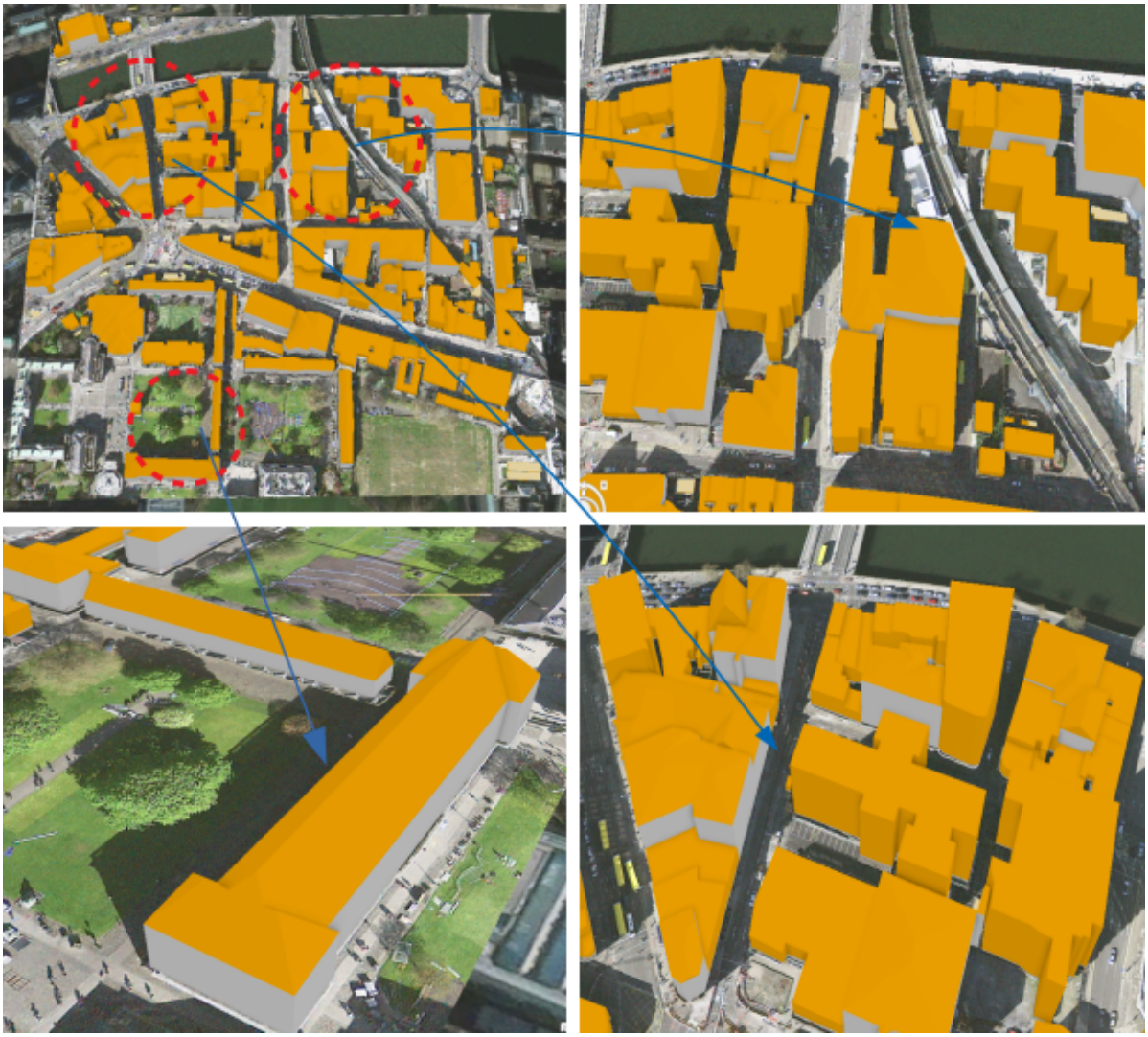


Figure 4.15: IM2ELEVATION inference results applied to roof profile estimation (LoD2).

In Fig. 4.15 we used ArcGIS functionalities to post-process the predicted DSM applying the footprint constraints (we use OSI-provided footprints data) via Local Government 3D Basemaps package[2]. The simple roof shape can be reconstructed to CityGML[53] style from 2.5D and reach the level of detail 2, i.e. LoD2. It is worth of mentioning that the work from Alidoost et al. [12] reached a similar level of precision, but they used additional data while training.

The training data used in the study covers solely urban areas of high density and lacks any examples of lower density or suburban scenes. Poor performance of our trained pipeline is therefore observed where buildings are sparsely distributed in the scene, see Fig. 4.16. This performance drop is due to the absence/weakness of support information for height inference, such as neighbouring buildings and shadows. The much stronger presence of tall vegetation also requires training of the pipeline on a more relevant suburban dataset.

[2]https://solutions.arcgis.com/local-government/help/local-government-scenes/get-started/

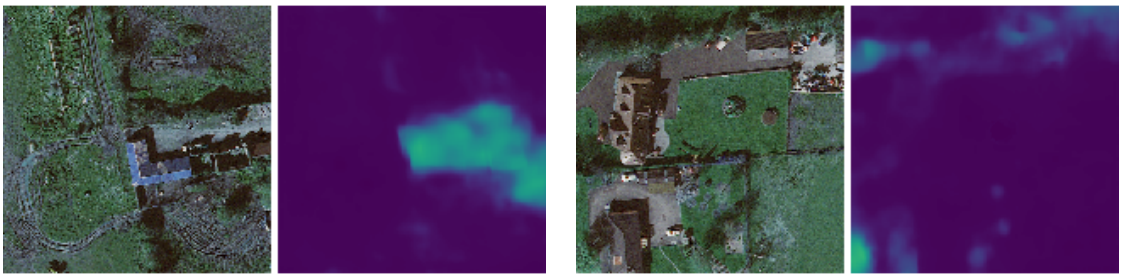


Figure 4.16: Examples of poor performance of DSM estimation on sparse scenes.

4.8 Conclusions

We have proposed a pipeline for inferring heights from aerial images using machine learning trained with two asynchronous heterogeneous datasets: aerial imagery and LiDAR point cloud, where the latter is employed only during the training phase. We mitigated the impact of temporal and spatial mismatches in our custom dataset by removing areas that had changed between the acquisition dates (e.g. new building, etc.), and also proposed registration tools on the two data streams to correct small translation artefacts. Our pipeline manages to perform building height estimation on average within 1.5 meters of the ground truth (which in this study is a LiDAR point cloud with an average height accuracy of 0.03 meters), which outperforms the estimation results on stereo imagery. We demonstrated how the estimated DSM could be used for inferring 3D building shapes and 2D building footprints.

Future research directions to enable higher accuracy in recovering heights and shapes of roofs include augmenting our training dataset with computer graphics simulated data (training set) and adding other input data streams such as street view imagery. IM2ELEVATION pipeline for DSM inference provides an alternative to stereo reconstruction when multiple aerial imageries are not available. Nevertheless, integration of the elements of this pipeline may provide highly relevant information in stereo reconstruction if such data is available. In the future, we envisage looking into merging multiple DSMs (i.e. stereo generated and AI-generated) to reduce noise and improve accuracy, in addition to considering shape priors inherent in man-built infrastructure (e.g. planar surface).

Chapter 5

Street view object geo-locating

5.1 Introduction

Monitoring public assets is a labour-consuming task, and for many decades, solutions collecting street view imagery have been routinely deployed in combination with computer vision-based approaches for object detection, and recognition in images [37].

Nowadays, street view images are available in massive amounts (e.g.: Mapillary[1], Google Street View (GSV)[2]) and additional information about the scene can be further extracted by machine learning techniques. Krylov et al. [71, 73] have employed deep learning modules for segmenting objects of interest (e.g. poles) in images and estimating their distance from the camera, and a Markov Random Field (MRF) is then used as a decision module to provide a usable list of the GPS coordinates of the assets of interest, limiting duplicates by reconciling detection from multiple view images.

The MRF conveniently merges information extracted from images and their metadata i.e. their associated camera location (GPS) and bearing information (cf. Fig. 1.4). Currently, the pipeline of Krylov et al. assumes that the metadata associated with the camera view pose is noiseless; however, it is not always the case (e.g. due to GPS receiver imprecision), and consequently, this noise affects the accuracy of the geo-location of the assets found. In this chapter, we propose to improve that pipeline by (1) denoising the camera metadata using Structure from Motion (SfM) and (2) using contextual information extracted from Open Street Map (OSM)[4] to push the predictions to a more probable area where the objects should be situated based on road and building locations. Fig. 5.1 summarizes our contributions and our approach has been validated for traffic light geolocation (c.f. Sec.5.4).

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

[2] https://developers.google.com/maps/documentation/streetview/overview

[3] https://play.google.com/store/apps/details?id=com.glidelinesystems.dioptra

[4] https://www.openstreetmap.org/

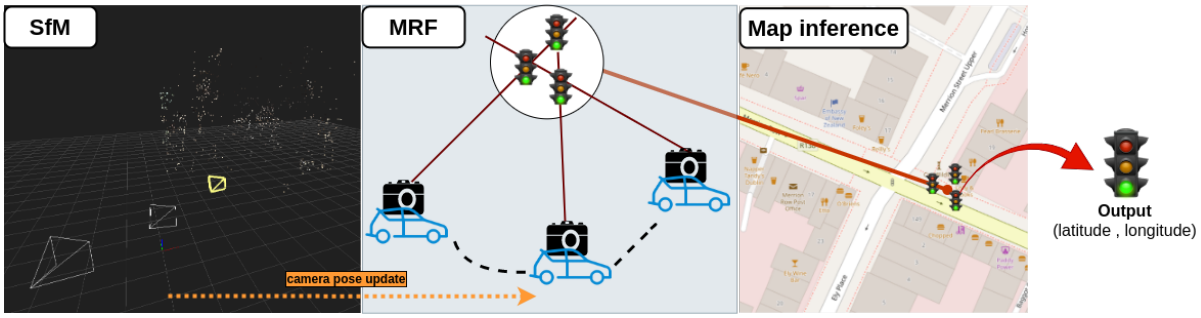


Figure 5.1: Pre-processing: SfM aims to de-noise camera metadata (i.e. poses) used as an input of the MRF. Post-processing: the Map prior module refines the result from MRF using contextual information from OSM.

5.2 The state of the art

5.2.1 Camera geolocalization

Various Simultaneous Localisation and Mapping (SLAM) and SfM techniques have been proposed to infer 3D points and to estimate the motion from a set of images [68, 128, 36, 80, 59]. Bundle adjustment (BA) is integrating matched points within a sequence of images and finding a solution simultaneously optimal with respect to both camera parameters and 3D points. Agarwal et al. [7] is the first to propose the bundle adjustment that is used in the structure from motion. The trajectory of camera pose estimation is based on relative measurements; error accumulation over time thus leads to drift. Lhuillier [81] proposed to use GPS geo-tag in the bundle adjustment optimisation. A similar problem is the camera re-localization [136, 6]. A GPS tag and SfM technique are used to geo-localise a street view image by estimating its relative pose against images from a database. Bresson et al. [26] and Kendall et al. [66] proposed to employ CNN (Convolution Neural Network) features to estimate camera pose transformation.

5.2.2 Object geotagging

Qin et al. [107, 106] proposed to estimate the instance-level depth of objects in images as an alternative to pixel-wise depth estimation. They found out that the latter (obtained by minimising the mean error for all pixels) sacrifices the accuracy of certain local areas in images. Bertoni et al. [20] employed prior knowledge of the average height of humans to perform pedestrian localisation. Qu et al. [108] proposed to detect and locate traffic signs from a monocular video stream. They relied on bundle adjustment with an image GPS geo-tag to reconstruct a sparse point cloud as a 3D map, then align it with several landmarks from the 3D city model generated by Soheilian et al. [120].

Wegner et al. [132] proposed a probabilistic model to locate trees. They employed multiple

modalities, including aerial view semantic segmentation, street view detection, and map information as well as the tree distance prior. Information is fused into a conditional random field (CRF) to predict the positions of trees. However, identical features may be mismatched in case the recurring objects sit nearby. To solve this issue, Nassar et al. [96, 97] employed the soft geometry constraint on geo-location of camera pose to identify the same object that appears in two views. They concatenate camera pose information together with image features and decode them using a CNN. The same object in the first view can be re-identified in the second view.

Nassar et al. [95] extend the method by constructing a graph from detected bounding boxes across the multi-views, feed the graph to a GNN [67] and let the GNN identify the same objects across different views. Hebbalaguppe. et al. [58] predicted bounding boxes around street objects, which was followed by the two-view epipolar constraint to reconstruct 3D feature points from the two observed scenes. However, the 3D feature point does not necessarily fall inside the target bounding box. Krylov et al. [73] employed the camera pose from multiple views as a soft constraint and used semantic segmentation of images alongside a monocular depth estimator to extract the information (bearing and depth) about objects of interest and feed the obtained information into an MRF model that predicts their locations.

5.2.3 Critical Analysis

The related literature outlined in Section. 5.2.1 deals with how to achieve accurate localisation of camera views. The fundamental ideas are mainly to rely on tracking image features to calculate the relative camera motions.

In Section. 5.2.2, the literature shows how to locate the object of interest given the camera localisation. Without having precise camera localisations, the prediction of objects may be imprecise. To tackle the geo-tagging problems, we attempt to develop a pipeline which deals with both issues to optimise the final prediction.

5.3 Method

We present camera calibration using the SfM technique in Section 5.3.1, which provides higher quality information to be used as an input to the MRF presented in Section 5.3.3. Section 5.3.4 proposes a post-processing method to refine the MRF predictions.

Figure 5.2: The top row image is the panorama image that covers 360 degrees horizontally. The second row is the rectilinear views split from the panorama image with the size of 640x640. We leverage deep learning modules to detect and locate the object in an image. Semantic segmentation (in Fig. row 4) [105] is used to detect the traffic lights in the scene. The available pre-train model is trained by the Mapillary Vistas dataset. Single depth estimation [51](in Fig. row 3) is employed to approximately predict the object's relative distance to the camera. The available pre-train model is trained by the KITTI dataset.

5.3.1 Structure from Motion: using optical observation to denoise on GPS data

The input represents a set of N panoramic street view images (360° field of view) captured with their metadata in an area of interest. Accurate camera geo-location is a key to accurately geo-locate objects in the scene. The GPS position in the metadata is inherently noisy, which lowers the accuracy of predicting object positions. To get a better estimate of the GPS coordinates associated with each camera position, we propose to tune each of the camera positions with a conventional 3D reconstruction pipeline [14], followed by bundle adjustment [7]. To ease image matching, we split the 360° panorama views into 8 overlapping rectilinear views: each view covers a 90-degree field of view and is overlapped by 45 degrees in the horizontal direction. Each view is then considered as an image captured by a pinhole camera, free of distortion (see Figure 5.2).

We aim to find all possible matching features extracted from our images and perform camera calibration to adjust the camera pose from image metadata.

We note the set of rectilinear views $\mathcal{V} = \{v_1^{(i)}, \cdots, v_8^{(i)}\}_{i=1,\cdots,N}$ where $v_1^{(i)}, \cdots, v_8^{(i)}$ corresponds to rectilinear views associated with panorama $i = 1, \cdots, N$ ($N = 112$ in our experiment). Suppose two views are matched by their detected features. Epipolar constraint with 5 point algorithm [14] is applied to find the essential matrix E, which establishes the geometry relationship between two views. E can be further decomposed into translation and rotation matrix, noted as R and τ, respectively. They can be put together as a transformation matrix $\Theta \in SE(3)$

$$\Theta = \begin{pmatrix} \mathrm{R} & \tau \\ 0 & 1 \end{pmatrix} \in \mathbb{R}^{4\times4} \quad \text{with} \quad \mathrm{R} \in SO(3) \text{ and } \tau \in \mathbb{R}^3. \tag{5.1}$$

Each calibrated view in $\mathcal{V}$ is associated with $\Theta = (\mathrm{R}, \tau)$, and these parameters can be estimated by minimising the re-projection error from 3D feature space to 2D image plane within a bundle of images.

Figure 5.3: The black dots are shown as the centre of panoramic views captured by Google car, whereas the green dots are that of the rectilinear view. Some of the rectilinear views are shifted to the centre of the panoramic view as those poses are optimised in bundle adjustment.

The Incremental reconstruction pipeline [80] is applied to automatically grow in searching all possible camera views to calibrate. As can be seen from Figure. 5.3, some cameras shift from the centre of the panorama after calibration. Note that we initialise with several pairs of views, which generate multiple routes of reconstructions. The reconstruction result reports are in Table. 5.1.

Many camera views are not added into reconstruction due to either difficulty establishing the epipolar geometry constraint or failure to minimise the re-projection error in the bundle adjustment process.

The images tracking result from SfM		
#GSV Images	#Tracking pairs	#GSV image rejected
896	692	464

Table 5.1: The tracking result from SfM. The number of tracking pairs suggests that the number of succeeded pairs is used in the reconstruction. Nearly half of the images are rejected during reconstruction

5.3.2 Preparation of data input for MRF model

To generate a ray pointing to the object, calculate the camera bearing as the direction of the ray. The bearing points to the centre of the image are noted as θ^0, and the image's field of view is 90. The bearing to the object can be inferred by the number of horizontal pixels away from the θ^0. Suppose an object is detected at the pixel (x, y) in semantic segmentation. The image width is w. The instance of an object's bearing θ can be obtained below:

$$\theta = \theta^0 + \frac{90}{w} \times \left(x - \frac{w}{2}\right) \tag{5.2}$$

We use the camera position p_c as the starting point of the ray. The magnitude of the ray d can be obtained by aligning the semantic map with the corresponding depth map at the same pixel (x, y). A ray is formatted as (p_c, θ, d) to input to the MRF model.

5.3.3 Object geo-location with MRF

The MRF model performs binary decisions on the nodes of a 2D graph, each node corresponding to an intersection between two rays. The rays correspond to rays (in 2D) with origins in the camera GPS coordinates and with directions of the bearings associated with the segmented object of interest (the pixel in the middle of the segmented object is chosen for the bearing information). In order to obtain the information, we employed two deep learning models within a pipeline. We first used the image segmentation model to segment the image for locating traffic lights in a 2D pixel coordinate. Second, we used the depth prediction model to estimate those segmented pixel distances from the camera. Using this pipeline (original from [73]), we can roughly locate the objects of interest in the world coordinate.

Each camera view provides one or many rays shooting to the objects of interest. The MRF model is optimised to perform a binary decision for each node concerning its occupancy by the object of

interest (i.e. 0 = no object, 1 = object present). For more information, please refer to [73]. Our contribution in this chapter is in providing more accurate GPS coordinates for the camera positions (than originally available in the image metadata) thanks to SfM, hence improving the geo-location of the nodes on this MRF and ultimately improving the accuracy of GPS coordinates for the objects of interest.

5.3.4 Post-processing

Because of the inaccuracies of the rays that define the MRF nodes, the same object may be associated with multiple nodes (Fig. 5.4 left) located in the same vicinity on the MRF graph. To resolve this issue, Krylov et al. [73] added a hierarchical clustering step after optimising the MRF to merge close positive sites together. The final position is the average of sites in the cluster. However, we have

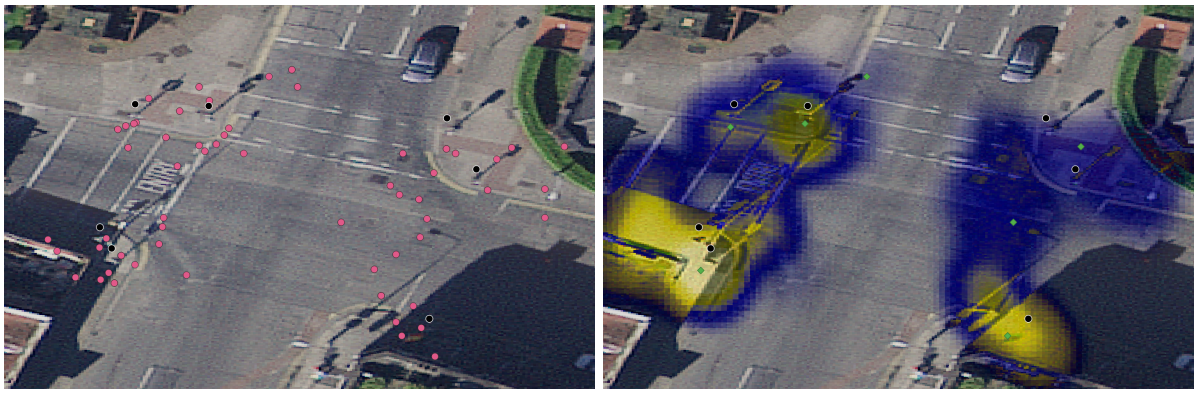


Figure 5.4: On the left figure, the red dots are positive intersections from MRF. The location of ground truth are shown in black dots. In the right figure, the green dots are the result of clustering process. The probability density function is applied to demonstrate the points' density of the intersections. The colour code from blue to yellow means the number of intersections from small to many.

observed that some positive sites were situated at improbable areas, for example, in the middle of the road. Therefore, we propose here to use OSM data to act as a useful prior for an area. As our objects of interest (e.g. traffic lights) are static and are located on the side of the road, we apply the following rule: *the object of interest can not be located in the middle of the road or around the edge of a building*. The OSM data has building and road classes represented by polygons and lines, respectively. A Normal kernel is fitted at each OSM node (cf. Fig. 5.5). Suppose a cluster C containing n positive sites $C = [c_1, c_2, ..., c_n]$, $W(z)$ is the function to query the weight that corresponds to the particular site in C and depends on the OSM nodes N_z within the proximity of the site z. The position P (equation 5.3) can be refined using a weighted average where certain sites

are penalised with small weights.

$$W(z) = 1 - \min\left(1, \sum_{\mu,\sigma \in N_z} -\exp\left(\frac{-(z-\mu)^2}{2\sigma^2}\right)\right) \quad \text{and} \quad P = \frac{\sum_{i=1}^{n} W(c_i) \times c_i}{\sum_{i=1}^{n} W(c_i)} \tag{5.3}$$

The σ stands for the standard deviation in meters and the μ is the node centre obtained from the OSM data. The σ is set to 2 meters for roads and 1 meter for building edges. The resolution of the Gaussian grid is 25 centimetres.

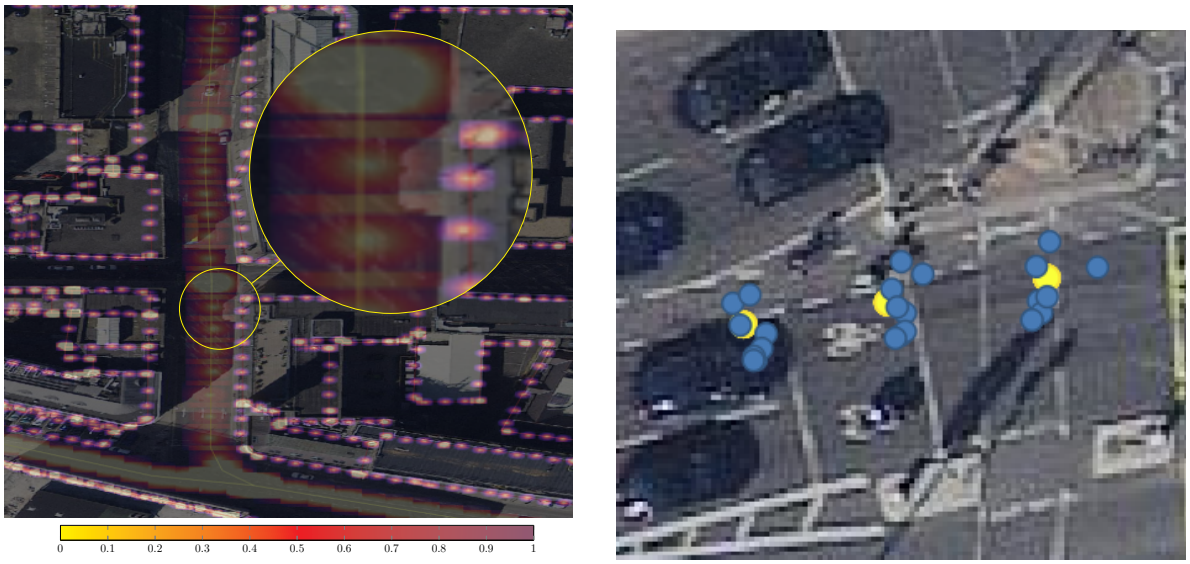


Figure 5.5: The figure (**left**) shows the prior map information overlaid on an aerial image. The normal kernel is applied to each node that is imported from OSM. The heatmap outlines improbable object locations that will have a smaller contribution to the weighted sum. On the (**right**) yellow dots are the positions taken from image metadata, and the blue dots represent their corrected versions with SfM.

5.3.5 Implementation

SURF descriptors [16] are used and in each view, 6,000 features are extracted. We employ FLANN (Fast Library for Approximate Nearest neighbours [94]) to match the SURF features between rectilinear views. RANSAC is used to remove outliers. We use the OpenSfM [5] to calibrate camera poses and Crese [7] as our solver to optimize the Θ. As the nodes from raw OSM data are not equally distributed, we interpolate the nodes every 5 meters in QGIS [6] to get a dense distribution of the map prior.

[5] https://github.com/mapillary/OpenSfM
[6] https://www.qgis.org/en/site/

The process efficiency could be quadratic($\mathcal{O}(n^2)$) as each image would try to match against the rest of the images, which is a time-consuming process. To speed the matching process, we constrain the maximum matching distance as 20 meters between images to reduce the unnecessary computation. To this end, the camera tracks are created according to these matching views.

5.4 Results

To validate our approach, we have used 896 GSV images (112 panoramas split into eight images each) collected in the Dublin city centre. The object of interest corresponds to a traffic light. We fine-tuned the input camera poses using the SfM (cf. Fig. 5.5). Our method corrected the bearing and position information on average by 4.36 degrees and 0.71 meters, respectively. Moreover, the use of the OSM prior results in an average refinement of the prediction by 0.17 meters.

#Ground truth	#Detected	TP	Precision↑	Recall↑	F-measure↑	Geo-localization error↓	Geo-localization error↓(with OSM)
no correction [73]							
76	94	58	0.61	0.76	0.68	2.71	2.64
correction on τ only							
76	89	57	0.64	0.75	0.69	2.79	2.74
correction on R and τ							
76	92	54	0.57	0.72	0.64	2.53	2.48

Table 5.2: We evaluate the impact of metadata correction by comparison with results that do not use any pose correction. By correcting the full camera pose (R and τ), the geo-location accuracy reaches an error of around 2.5 meters to a reference point. It outperforms the result with no correction by 18 cm and 16 cm after applying the OSM prior. We reach the highest F-measure if only the τ is corrected.

By using our SFM module, we can check the impact of the following correction of the metadata: correction on τ only (i.e. GPS location of the camera), correction on R and τ (i.e. correction of both GPS location and bearing of the camera). To validate our approach, we use the original metadata as our baseline for comparisons. Table 5.2 shows the testing results in terms of geo-localisation error and precision and recall detection metrics. We consider traffic lights to be recovered accurately (true positive) if they are located within 6 meters from the reference position; otherwise it is viewed as a false positive. The geo-localisation error measures the average Haversine distance between the prediction and its reference target in meters. A small distance indicates accurate position prediction.

We compare our results with related public asset geo-location approaches in Table 5.3. The pro-

posed technique reaches the smallest positional error; however, the results are not directly comparable due to the different complexity of the scene and detected objects.

Fig. 5.6 shows one of the examples for the difference between no correction and correction on R and τ. The prediction with correction is closer to its reference position. We import the result into Google Earth [7] and visualize them in 3D (see Fig. 5.6).

Comparison with other methods			
Method	Dataset	F-measure↑	Geo-localization error↓
Siamese CNN [97]	Pasadena [132]	0.51	3.13
Siamese CNN	Mapillary [98]	0.72	4.36
GNN-Geo [95]	Pasadena	0.64	2.75
GNN-Geo	Mapillary	0.87	4.21
Ours	**DTL** [73]	**0.64**	**2.48**

Table 5.3: In comparison with other approaches, our method achieves the smallest geo-localisation error, although the other datasets might be more challenging for object detection.

5.5 Conclusion

We have shown that by denoising metadata associated with street view imagery using SfM and by using context information such as road and building shapes extracted from OSM, assets of interest can be geolocalised with higher accuracy. Currently, our pipeline is geo-tagging one class of objects at a given time, and future work will investigate multiple static object class tagging with additional priors associated with their relative positioning in the scene, to improve further geo-location accuracy.

[7] https://www.google.com/earth/

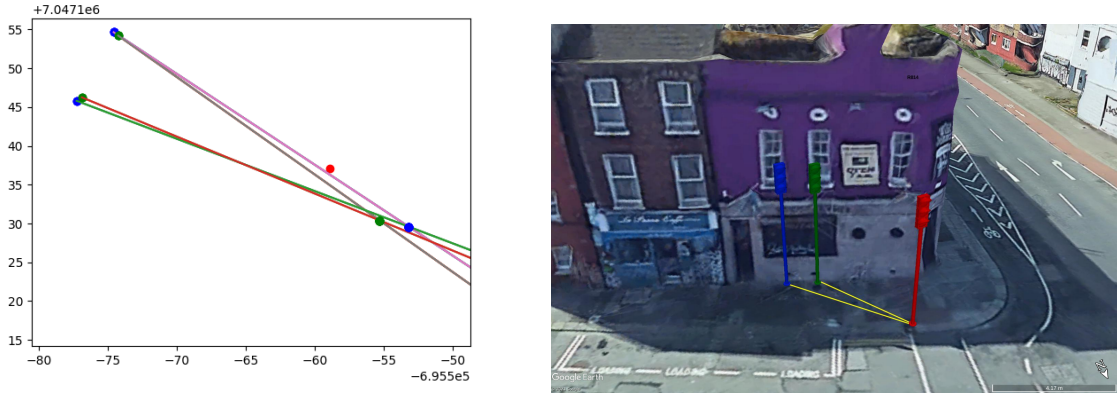


Figure 5.6: The left figure shows the positive ray intersection between the two. The red dot stands for ground truth position. The blue pairs are the camera pose with no correction, and the green pairs are the camera pose with correction. Their pair of rays intersect at the position with the same colour corresponding to their camera. As can be seen, the intersection of the green dot is closer to the ground truth. The right figure shows the position of intersection and ground truth in Google Earth in the same colour scheme.

Chapter 6

Conclusion and future work

At the end of the dissertation, we conclude the thesis by summarizing the contributions followed by revisiting the research question and discussing some of the potential research directions.

6.1 Summary

We studied different aspects of creating a better dataset for the machine to train in remote sensing applications and proposed deep learning-based solutions. We are dedicated to measuring how much better the performance of the machine can be improved by learning through better training examples. The validation has been made through the following applications in remote sensing and in street view.

- The impact of denoising annotation for deep learning prediction.
- The impact of denoising GPS data for object geotagging from street view.

In Chapter 3 we attempt to fuse geographic information from a 2D GIS layer with point cloud data, performing the 3D semantic segmentation and GIS polygonal label. We approach the problem by identifying building units and adjusting the polygonal perimeters from GIS to fit with the point cloud. This pipeline generates a more meaningful annotation from GIS data, which potentially can be used for more accurately pairing with aerial images. This result makes us investigate more into creating annotations for machine learning applications.

However, the point cloud created by flying a drone over the region is limited in the scalability of collecting imagery and is prone to have georeferencing errors in imagery (please refer to Sec. 2.1.1). If one wanted to have a larger collection of imagery and more accurate georeference, we explore the point cloud data collected by aerial LiDAR platform with orthographic aerial imagery that captured the same region at different times. This combination gives us a larger collection. Moreover, the orthophoto provides us with a higher resolution and better georeference.

Despite mapping two sources into the same coordinate of georeferencing, discrepancies occurred between these two sources of data. Directly letting the machine learn from this pair of data may confuse it while training. We propose a pipeline **??** to mitigate the impact of temporal and spatial mismatches by automatically registering the pair of asynchronous heterogeneous datasets. In Chapter 4, we validate the performance of preprocessing dataset and propose an end-to-end CNN architecture to map the aerial imagery to the DSM. Comparing the same task with other proposed datasets, our dataset is comparable to those carefully calibrated synchronous datasets.

This compelling finding shows that historical data can be reused to perform a similar outcome in comparison with the carefully synchronized dataset. Our proposed deep learning architecture reaches the state of the art in accuracy with an margin of error less than 1.5 metres estimating height from a single aerial image. To this end, we are able to use the result to reconstruct roofs of buildings.

While the aerial view can cover a larger scale of an area by focusing on the structure of the city or geographical landscape, the street view serves as discovering the objects which appeared in the local scene. In Chapter 5, we look into the GPS error that associates each street view imagery and use the SfM to denoise the error. We validate the corrected GPS data by testing in geotagging traffic light and refine the prediction by the map information from OSM. We achieve better results in traffic light detection and geolocation error with our data processing pipeline.

6.2 Future work

This thesis introduces several novel ideas about using multi-modality of data for preparing deep/machine learning dataset. In this final section, we will outline some interesting directions for refinement or application that could be worth investigating further.

Aerial view colour correction: we found the aerial imagery from different datasets may have different intensities in RGB channels, which leads the trained model failing in predicting the DSM from colour images (see Figure. 4.14). The solution can be as easy as normalizing the input image, however, it does not appear to work in our case. The question could be how can we transfer each source of the image to a target before feeding the image to the trained model. Colour transferring(see Figure. 6.1) could be a preprocessing step to diminish the difference in colour from different datasets. Pitie et al. [104] leveraged on probability density function (PDF) to estimate the transformation of the colour from source to target. The estimation process efficiently in 1D mapping to minimize the colour differences. Alghamdi et al. [10][11] proposed to function mapping the source image to target by analysing on local patch colour distribution. This minimizes the colour variations and potentially makes the image adaptable to the trained model.

Figure 6.1: Left: the input images from Google Earth. Middle: the corresponding area from OSI [83]. Right: the images processed by colour transferring function in [104]. The goal is to transfer the colour of the left image as similar as the middle one so that the trained model is able to make accurate predictions.

Bibliography

[1] 2018 IEEE GRSS Data Fusion Contest. Online: http://www.grss-ieee.org/community/technical-committees/data-fusion [Last accessed: 24 July 2020].

[2] HERE Geodata Models offer global precise 3D dataset for 5G deployment. *Geo Week News*, 2 April 2020. Available: https://www.geo-week.com/here-geodata-models-offer-global-precise-3d-dataset-for-deploying-5g/ [Last accessed: 24 July 2020].

[3] ISPRS Potsdam 2D Semantic Labeling Contest. Online: http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-potsdam.html [Last accessed: 24 July 2020].

[4] ISPRS Vaihingen 2D Semantic Labeling Dataset. Online: http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-vaihingen.html [Last accessed: 24 July 2020].

[5] LIDAR Point Cloud UK. https://data.gov.uk/dataset/977a4ca4-1759-4f26-baa7-b566bd7ca7bf/lidar-point-cloud [Last accessed: 24 July 2020].

[6] Pratik Agarwal, Wolfram Burgard, and Luciano Spinello. Metric localization using google street view. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3111–3118. IEEE, 2015.

[7] Sameer Agarwal, Noah Snavely, Steven M Seitz, and Richard Szeliski. Bundle adjustment in the large. In *European conference on computer vision*, pages 29–42. Springer, 2010.

[8] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M Seitz, and Richard Szeliski. Building rome in a day. In *2009 IEEE 12th international conference on computer vision*, pages 72–79. IEEE, 2009.

[9] Kashif Ahmad, Konstantin Pogorelov, Michael Riegler, Olga Ostroukhova, Pål Halvorsen, Nicola Conci, and Rozenn Dahyot. Automatic detection of passable roads after floods in

remote sensed and social media data. *Signal Processing: Image Communication*, 74:110–118, 2019.

[10] Hana Alghamdi and Rozenn Dahyot. Iterative nadaraya-watson distribution transfer for colour grading. In *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6. IEEE, 2020.

[11] Hana Alghamdi and Rozenn Dahyot. Patch based colour transfer using SIFT flow. In *Irish Machine Vision and Image Processing (IMVIP 2020)*, volume abs/2005.09015, 2020. Best Paper Award, Book Open Access http://research.thea.ie/handle/20.500.12065/3429.

[12] Fatemeh Alidoost, Hossein Arefi, and Federico Tombari. 2d image-to-3d model: Knowledge-based 3d building reconstruction (3dbr) using single aerial images and convolutional neural networks (cnns). *Remote Sensing*, 11(19):2219, 2019.

[13] Hamed Amini Amirkolaee and Hossein Arefi. Height estimation from single aerial images using a deep convolutional encoder-decoder network. *ISPRS journal of photogrammetry and remote sensing*, 149:50–66, 2019.

[14] Alex M Andrew. Multiple view geometry in computer vision. *Kybernetes*, 2001.

[15] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:2481–2495, 2017.

[16] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[17] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[18] C. Becker, N. Häni, E. Rosinskaya, E. d'Angelo, and C. Strecha. Classification of aerial photogrammetric 3d point clouds. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-1/W1:3–10, 2017.

[19] C. Benedek, X. Descombes, and J. Zerubia. Building development monitoring in multitemporal remotely sensed image pairs with stochastic birth-death dynamics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):33–50, 2012.

[20] Lorenzo Bertoni, Sven Kreiss, and Alexandre Alahi. Monoloco: Monocular 3d pedestrian localization and uncertainty estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6861–6871, 2019.

[21] Ksenia Bittner, Pablo d'Angelo, Marco Körner, and Peter Reinartz. Dsm-to-lod2: Spaceborne stereo digital surface model refinement. *Remote Sensing*, 10(12):1926, 2018.

[22] A. Bódis-Szomorú, H. Riemenschneider, and L. Van Gool. Efficient volumetric fusion of airborne and street-side data for urban reconstruction. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 3204–3209, Dec 2016.

[23] Marc Bosch, Kevin Foster, Gordon Christie, Sean Wang, Gregory D Hager, and Myron Brown. Semantic stereo for incidental satellite images. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1524–1532. IEEE, 2019.

[24] Christian Boucheny. *Visualisation scientifique de grands volumes de données: Pour une approche perceptive*. PhD thesis, 2009.

[25] Steve Branson, Jan Dirk Wegner, David Hall, Nico Lang, Konrad Schindler, and Pietro Perona. From google maps to a fine-grained catalog of street trees. *ISPRS Journal of Photogrammetry and Remote Sensing*, 135:13 – 30, 2018.

[26] Guillaume Bresson, Li Yu, Cyril Joly, and Fabien Moutarde. Urban localization with street views using a convolutional neural network for end-to-end camera pose regression. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1199–1204. IEEE, 2019.

[27] Abdullah Bulbul and Rozenn Dahyot. Social media based 3d visual popularity. *Computers & Graphics*, 63:28 – 36, 2017.

[28] J. Byrne, J. Connelly, J. Su, V. Krylov, M. Bourke, D. Moloney, and R. Dahyot. Trinity college dublin drone survey dataset. Technical report, School of Computer Science & Statistics, Trinity College Dublin, 2017.

[29] J. Byrne, D. F. Laefer, and E. O'Keeffe. Maximizing feature detection in aerial unmanned aerial vehicle datasets. *Journal of Applied Remote Sensing*, 11(2):025015, April 2017.

[30] Marcela Carvalho, Bertrand Le Saux, Pauline Trouvé-Peloux, Frédéric Champagnat, and Andrés Almansa. Multitask learning of height and semantics from aerial images. *IEEE Geoscience and Remote Sensing Letters*, 2019.

[31] Mohamed Chaabane, Lionel Gueguen, Ameni Trabelsi, Ross Beveridge, and Stephen O'Hara. End-to-end learning improves static object geo-localization from video. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2063–2072, 2021.

[32] Honglie Chen, Weidi Xie, Andrea Vedaldi, and Andrew Zisserman. Autocorrect: Deep inductive alignment of noisy geometric annotations. *arXiv preprint arXiv:1908.05263*, 2019.

[33] Honglie Chen, Weidi Xie, Andrea Vedaldi, and Andrew Zisserman. Autocorrect: Deep inductive alignment of noisy geometric annotations. 2019.

[34] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[35] Gabriela Csurka, Diane Larlus, Florent Perronnin, and France Meylan. What is a good evaluation measure for semantic segmentation? In *BMVC*, volume 27, 2013.

[36] Mark Cummins. Highly scalable appearance-only slam-fab-map 2.0. *Proc. Robotics: Sciences and Systems (RSS), 2009*, 2009.

[37] Rozenn Dahyot. *Analyse d'images séquentielles de scènes routières par modèles d'apparence pour la gestion du réseau routier (Appearance based road scene video analysis for the management of the road network)*. PhD thesis, University of Strasbourg I, France, November 2001. (published in French).

[38] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[39] Michal Drozdzal, Eugene Vorontsov, Gabriel Chartrand, Samuel Kadoury, and Chris Pal. The importance of skip connections in biomedical image segmentation. In *Deep Learning and Data Labeling for Medical Applications*, pages 179–187. Springer, 2016.

[40] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015.

[41] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[42] Gabriele Facciolo, Carlo De Franchis, and Enric Meinhardt-Llopis. Automatic 3d reconstruction from multi-date satellite images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 57–66, 2017.

[43] Steven Fortune. A sweepline algorithm for voronoi diagrams. *Algorithmica*, 2(1-4):153, 1987.

[44] P Fricker and A Rohrbach. Pushbroom scanners provide highest resolution earth imaging information in multispectral bands. In *Proceedings of IPSRS Hannover Workshop: High Resolution Earth Imaging for Geospatial Information*, 2005.

[45] Francisco García-Sánchez, Luis Galvez-Sola, Juan J Martínez-Nicolás, Raquel Muelas-Domingo, and Manuel Nieves. Using near-infrared spectroscopy in agricultural systems. *Kyprianidis KG, Skvaril J. Developments in near-infrared spectroscopy. London: IntechOpen*, pages 97–127, 2017.

[46] Pedram Ghamisi and Naoto Yokoya. Img2dsm: Height simulation from single imagery using conditional generative adversarial net. *IEEE Geoscience and Remote Sensing Letters*, 15(5):794–798, 2018.

[47] Nicolas Girard, Guillaume Charpiat, and Yuliya Tarabalka. Aligning and updating cadaster maps with aerial images by multi-task, multi-resolution deep learning. In *Asian Conference on Computer Vision*, pages 675–690. Springer, 2018.

[48] Ross B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015.

[49] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[50] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017.

[51] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3828–3838, 2019.

[52] Samuel N Goward, Jeffrey G Masek, Darrel L Williams, James R Irons, and RJ Thompson. The landsat 7 mission: Terrestrial research and applications for the 21st century. *Remote Sensing of Environment*, 78(1-2):3–12, 2001.

[53] Gerhard Gröger and Lutz Plümer. Citygml–interoperable semantic 3d city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 71:12–33, 2012.

[54] Mads Haahr. Creating location-based augmented-reality games for cultural heritage. In Mariano Alcañiz, Stefan Göbel, Minhua Ma, Manuel Fradinho Oliveira, Jannicke Baalsrud Hauge, and Tim Marsh, editors, *Serious Games*. Springer International Publishing, 2017.

[55] Ayman Habib, Mwafag Ghanma, Michel Morgan, and Rami Al-Ruzouq. Photogrammetric and lidar data registration using linear features. *Photogrammetric Engineering & Remote Sensing*, 71(6):699–707, 2005.

[56] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[57] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[58] Ramya Hebbalaguppe, Gaurav Garg, Ehtesham Hassan, Hiranmay Ghosh, and Ankit Verma. Telecom inventory management via object recognition and localisation on google street view images. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 725–733. IEEE, 2017.

[59] Jared Heinly, Johannes L Schonberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the world* in six days*(as captured by the yahoo 100 million image dataset). In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3287–3295, 2015.

[60] Heiko Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 807–814. IEEE, 2005.

[61] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[62] Junjie Hu, Mete Ozay, Yan Zhang, and Takayuki Okatani. Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1043–1051. IEEE, 2019.

[63] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[64] Pascal Kaiser, Jan Dirk Wegner, Aurélien Lucchi, Martin Jaggi, Thomas Hofmann, and Konrad Schindler. Learning aerial image segmentation from online maps. *IEEE Transactions on Geoscience and Remote Sensing*, 55(11):6054–6068, 2017.

[65] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006.

[66] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.

[67] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[68] Bryan Klingner, David Martin, and James Roseborough. Street view motion-from-structure-from-motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 953–960, 2013.

[69] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[70] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[71] V. Krylov and R. Dahyot. Object geolocation using mrf-based multi-sensor fusion. In *Proc. of IEEE Int'l Conf on Image Processing ICIP*, Athens, Greece, 2018.

[72] Vladimir A. Krylov, Eamonn Kenny, and Rozenn Dahyot. Automatic discovery and geotagging of objects from street view imagery. *Remote Sensing*, 10(5), 2018.

[73] Vladimir A. Krylov, Eamonn Kenny, and Rozenn Dahyot. Automatic discovery and geotagging of objects from street view imagery. *Remote Sensing*, 10(5), 2018.

[74] Tae-Suk Kwak, Yong-Il Kim, Ki-Yun Yu, and Byoung-Kil Lee. Registration of aerial imagery and aerial lidar data using centroids of plane roof surfaces as control information. *KSCE journal of Civil Engineering*, 10(5):365–370, 2006.

[75] D. F. Laefer, S. Abuwarda, A.-V. Vo, L. Truong-Hong, and H. Gharibi. 2015 aerial laser and photogrammetry survey of dublin city collection record. Technical report, New York University, 2017. 3D point cloud dataset. `https://doi.org/10.17609/N8MQ0N`.

[76] Debra F. Laefer, Anh-Vu Vo Saleh Abuwarda, Linh Truong-Hong, and Hamid Gharibi. 2015 aerial laser and photogrammetry survey of dublin city collection record. https://doi.org/10.17609/N8MQ0N, 2015. Corresponding dataset available at `https://geo.nyu.edu/catalog/nyu_2451_38684`.

[77] F. Lafarge, X. Descombes, J. Zerubia, and M. Pierrot-Deseilligny. Structural approach for building reconstruction from a single dsm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):135–147, 2010.

[78] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016.

[79] Daniel Laumer, Nico Lang, Natalie van Doorn, Oisin Mac Aodha, Pietro Perona, and Jan Dirk Wegner. Geocoding of trees from street addresses and street-level images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 162:125 – 136, 2020.

[80] Taehee Lee. Robust 3d street-view reconstruction using sky motion estimation. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 1840–1847. IEEE, 2009.

[81] Maxime Lhuillier. Fusion of gps and structure-from-motion using constrained bundle adjustments. In *CVPR 2011*, pages 3025–3032. IEEE, 2011.

[82] Chao-Jung Liu, Vladimir Krylov, and Rozenn Dahyot. 3d point cloud segmentation using gis. In *20th Irish Machine Vision and Image Processing Conference*, pages 41–48, 2018.

[83] Chao-Jung Liu, Vladimir A Krylov, Paul Kane, Geraldine Kavanagh, and Rozenn Dahyot. Im2elevation: Building height estimation from single-view aerial imagery. *Remote Sensing*, 12(17):2719, 2020.

[84] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5162–5170, 2015.

[85] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[86] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987.

[87] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[88] Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2017.

[89] Fangchang Mal and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8. IEEE, 2018.

[90] Jordan M Malof, Kyle Bradbury, Leslie M Collins, and Richard G Newell. Automatic detection of solar photovoltaic arrays in high resolution aerial imagery. *Applied energy*, 183:229–240, 2016.

[91] Andrew Mastin, Jeremy Kepner, and John Fisher. Automatic registration of lidar and optical images of urban scenes. In *2009 IEEE conference on computer vision and pattern recognition*, pages 2639–2646. IEEE, 2009.

[92] B. Micusik and J. Kosecka. Piecewise planar city 3d modeling from street view panoramic sequences. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2906–2912, 2009.

[93] Lichao Mou and Xiao Xiang Zhu. Im2height: Height estimation from single monocular imagery via fully residual convolutional-deconvolutional network. *arXiv preprint arXiv:1802.10249*, 2018.

[94] Marius Muja and David G Lowe. Fast matching of binary features. In *2012 Ninth conference on computer and robot vision*, pages 404–410. IEEE, 2012.

[95] Ahmed Samy Nassar, Stefano D'Aronco, Sébastien Lefèvre, and Jan D Wegner. Geograph: Graph-based multi-view object detection with geometric cues end-to-end. In *European Conference on Computer Vision*, pages 488–504. Springer, 2020.

[96] Ahmed Samy Nassar, Nico Lang, Sébastien Lefèvre, and Jan D Wegner. Learning geometric soft constraints for multi-view instance matching across street-level panoramas. In *2019 Joint Urban Remote Sensing Event (JURSE)*, pages 1–4. IEEE, 2019.

[97] Ahmed Samy Nassar, Sébastien Lefèvre, and Jan Dirk Wegner. Multi-view instance matching with learned geometric soft-constraints. *ISPRS International Journal of Geo-Information*, 9(11):687, 2020.

[98] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4990–4999, 2017.

[99] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

[100] D. Palmer, E. Koumpli, I. Cole, R. Gottschalg, and T. A Betts. Gis-based method for identification of wide area rooftop suitability for minimum size pv systems using lidar data and photogrammetry. *Energies*, 11(12:3506), 2018.

[101] Ebadat G Parmehr, Clive S Fraser, Chunsun Zhang, and Joseph Leach. Automatic registration of optical imagery with 3d lidar data using statistical similarity. *ISPRS Journal of Photogrammetry and Remote Sensing*, 88:28–40, 2014.

[102] Shubiao Peng, Hongchao Ma, and Liang Zhang. Automatic registration of optical images with airborne lidar point cloud in urban scenes based on line-point similarity invariant and extended collinearity equations. *Sensors*, 19(5):1086, 2019.

[103] Hai Minh Pham, Yasushi Yamaguchi, and Thanh Quang Bui. A case study on the relation between city planning and urban growth using remote sensing and spatial metrics. *Landscape and Urban Planning*, 100(3):223–230, 2011.

[104] F. Pitie, A.C. Kokaram, and R. Dahyot. N-dimensional probability density function transfer and its application to color transfer. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1434–1439 Vol. 2, Oct 2005.

[105] Lorenzo Porzi, Samuel Rota Bulò, Aleksander Colovic, and Peter Kontschieder. Seamless scene segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[106] Zengyi Qin, Jinglu Wang, and Yan Lu. Monogrnet: A geometric reasoning network for monocular 3d object localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8851–8858, 2019.

[107] Zengyi Qin, Jinglu Wang, and Yan Lu. Triangulation learning network: from monocular to stereo 3d object detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[108] Xiaozhi Qu, Bahman Soheilian, and Nicolas Paparoditis. Vehicle localization using mono-camera and geo-referenced signs. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 605–610. IEEE, 2015.

[109] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

[110] Hayko Riemenschneider, András Bódis-Szomorú, Julien Weissenberg, and Luc Van Gool. Learning where to classify in multi-view semantic segmentation. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 516–532. Springer, 2014.

[111] Markus Rumpler, Arnold Irschara, Andreas Wendel, and Horst Bischof. *Rapid 3D City Model Approximation from Publicly Available Geographic Data Sources and Georeferenced Aerial Images*, pages 1–8. ., 2012.

[112] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. Learning depth from single monocular images. In *Advances in neural information processing systems*, pages 1161–1168, 2006.

[113] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Depth perception from a single still image. In *AAAI*, volume 3, pages 1571–1576, 2008.

[114] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[115] Heiko G Seif and Xiaolong Hu. Autonomous driving in the icity—hd maps as a key challenge of the automotive industry. *Engineering*, 2(2):159–162, 2016.

[116] Claude E Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.

[117] Jamie Shotton, Matthew Johnson, and Roberto Cipolla. Semantic texton forests for image categorization and segmentation. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[118] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *European conference on computer vision*, pages 1–15. Springer, 2006.

[119] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[120] Bahman Soheilian, Olivier Tournaire, Nicolas Paparoditis, Bruno Vallet, and Jean-Pierre Papelard. Generation of an integrated 3d city model with visual landmarks for autonomous navigation in dense urban areas. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 304–309. IEEE, 2013.

[121] X. Song, Y. Huang, C. Zhao, Y. Liu, Y. Lu, Y. Chang, and J. Yang. An approach for estimating solar photovoltaic potential based on rooftop retrieval from remote sensing images. *Energies*, 11(11:3172), 2018.

[122] P Srinivas, V Raghu Venkataraman, and I Jayalakshmi. Digital aerial orthobase for cadastral mapping. *Journal of the Indian Society of Remote Sensing*, 40(3):497–506, 2012.

[123] Shivangi Srivastava, Michele Volpi, and Devis Tuia. Joint height estimation and semantic labeling of monocular aerial images with cnns. In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 5173–5176. IEEE, 2017.

[124] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

[125] Martin Styner, Christian Brechbuhler, G Szckely, and Guido Gerig. Parametric estimate of intensity inhomogeneities applied to mri. *IEEE transactions on medical imaging*, 19(3):153–165, 2000.

[126] Lyne P. Tchapmi, Christopher B. Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3d point clouds. In *International Conference on 3D Vision (3DV)*, 2017.

[127] Joseph Tighe and Svetlana Lazebnik. SuperParsing: Scalable Nonparametric Image Parsing with Superpixels. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 352–365, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[128] Akihiko Torii, Michal Havlena, and Tomáš Pajdla. From google street view to 3d city models. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 2188–2195. IEEE, 2009.

[129] Dimosthenis C. Tsouros, Stamatia Bibi, and Panagiotis G. Sarigiannidis. A review on uav-based applications for precision agriculture. *Information*, 10(11), 2019.

[130] John E Vargas-Muñoz, Sylvain Lobry, Alexandre X Falcão, and Devis Tuia. Correcting rural building annotations in openstreetmap using convolutional neural networks. *ISPRS journal of photogrammetry and remote sensing*, 147:283–293, 2019.

[131] Paul Viola and William M Wells III. Alignment by maximization of mutual information. *International journal of computer vision*, 24(2):137–154, 1997.

[132] 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.

[133] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[134] Dan Xu, Elisa Ricci, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Multi-scale continuous crfs as sequential deep networks for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5354–5362, 2017.

[135] Yonghao Xu, Bo Du, Liangpei Zhang, Daniele Cerra, Miguel Pato, Emiliano Carmona, Saurabh Prasad, Naoto Yokoya, Ronny Hänsch, and Bertrand Le Saux. Advanced multi-sensor optical remote sensing for urban land use and land cover classification: Outcome of the 2018 ieee grss data fusion contest. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(6):1709–1724, 2019.

[136] Li Yu, Cyril Joly, Guillaume Bresson, and Fabien Moutarde. Monocular urban localization using street view. In *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 1–6. IEEE, 2016.

[137] F Yuan, JX Zhang, L Zhang, and J Gao. Dem generation from airborne lidar data. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38(7/C4):308–312, 2009.

[138] Armand Zampieri, Guillaume Charpiat, Nicolas Girard, and Yuliya Tarabalka. Multimodal image alignment through a multiscale chain of neural networks with application to remote sensing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 657–673, 2018.

[139] Chaoquan Zhang, Hongchao Fan, Wanzhi Li, Bo Mao, and Xuan Ding. Automated detecting and placing road objects from street-level images. *arXiv preprint arXiv:1909.05621*, 2019.

[140] Weixing Zhang, C. Witharana, Weidong Li, C. Zhang, Xiaojiang Li, and Jason Parent. Using deep learning to identify utility poles with crossarms and estimate their locations from google street view images. *Sensors (Basel, Switzerland)*, 18, 2018.

[141] Wuming Zhang, Jing Zhao, Mei Chen, Yiming Chen, Kai Yan, Linyuan Li, Jianbo Qi, Xiaoyan Wang, Jinghui Luo, and Qing Chu. Registration of optical imagery and lidar data using an inherent geometrical constraint. *Optics express*, 23(6):7694–7702, 2015.

[142] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2):119–152, 1994.

[143] Wei Zhuo, Mathieu Salzmann, Xuming He, and Miaomiao Liu. Indoor scene structure analysis for single image depth estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 614–622, 2015.

Appendix A

Abbreviations

Short Term	Expanded Term
GPS	Global Positioning System
OSM	Open Street Map
OSI	Ordnance Survey Ireland
GIS	Geographical Information System
GSD	Ground sample distance
SfM	Structure from Motion
CNN	Convolutional Neural Network
GPU	Graphics Processing Unit
RGB	Red Green Blue
PDF	Probability Density Function

Appendix B

Variables

Notation	Description
$\hat{T}$	Optimal translation for polygon shape
x, y	Pixel coordinates in image
S	Polygon shape - a list of 2D vectors
r	Radius in meter
H	The marginal (Shannon) entropy
p	Image intensity distribution
MI	Mutual information
R	Rotation matrix
$SO(3)$	Special orthogonal group in 3D
$\mathbb{R}^3$	3D Euclidean space
Θ	Special Euclidean group in 3D
θ	Bearing degree
w	Image width
p_c	Pixel at a coordinate (x , y)
d	Distance from a camera centre to an object