3D OBJECT DETECTION AND REPRESENTATION IN REMOTE SENSING: PROBABILISTIC METHODS AND APPLICATIONS

by

Abdullah Himmet Özcan

Submitted to Graduate School of Natural and Applied Sciences
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy in
Electrical and Electronics Engineering

Yeditepe University

2017

3D OBJECT DETECTION AND REPRESENTATION IN REMOTE SENSING:
PROBABILISTIC METHODS AND APPLICATIONS

APPROVED BY:

Prof. Dr. Cem Ünsalan (Thesis Supervisor)	[illegible]
Prof. Dr. Duygun Erol Barkana	[illegible]
Assist. Prof. Dr. Engin Maşazade	[illegible]
Assoc. Prof. Dr. Gökhan Bora Esmer	[illegible]
Assoc. Prof. Dr. Yusuf Sinan Akgül	[illegible]

DATE OF APPROVAL: .... /.... /2017

ACKNOWLEDGEMENTS

First and foremost, I would like to thank both personally and professionally, to my advisor Cem Ünsalan for his guidance, encouragement and friendship throughout my graduate study at Yeditepe University. I am also grateful to my cosupervisors Engin Maşazade and Fatih Kahraman for their support.

Finally, I would like to thank my family: my wife Hanife and my son Nejat Kağan for supporting me spiritually throughout writing this thesis.

ABSTRACT

3D OBJECT DETECTION AND REPRESENTATION IN REMOTE SENSING: PROBABILISTIC METHODS AND APPLICATIONS

Nowadays, satellite images and three dimensional data are actively used in various areas. The most important of these is the detection of objects after a natural disaster using satellite images or three dimensional data. In fact, this information is also valuable for government agencies, city and regional planners even if no natural disaster occurs. Turkey has its own remote sensing satellites in the orbit. There are also plans to launch new and advanced remote sensing satellites in the near future. Although we have our own remote sensing satellites in the orbit, these can only provide raw images. Either an operator should extract information from them or the information may be extracted by software automatically. The first option is not applicable most of the times since the size of the raw images are huge. Also, the objects to be detected from them are tiny compared to the image size. In this thesis, novel methods for object detection and segmentation are proposed. Remote sensing objects are detected using combinations of local features and shapes in a novel probabilistic voting framework. The shape of the objects are extracted using satellite images and height data. First, we developed a novel back-projection method to obtain the shape of detected objects in satellite images. Then for height data, two novel segmentation and filtering methods are proposed. The first method depends on the probabilistic voting method with a novel morphological based region growing algorithm. The second method uses empirical mode decomposition (EMD) algorithm for filtering and segmenting DSM into ground and non-ground points. The proposed methods are tested on different satellite images (IKONOS, WorldView, QuickBird,) and three dimensional data (DSM, LIDAR). Compared with the methods in the literature, better results have been obtained.

ÖZET

UZAKTAN ALGILAMADA 3D NESNE TESPİTİ VE TEMSİLİ: OLASILIKSAL YÖNTEMLER VE UYGULAMALAR

Günümüzde uydu görüntüleri ve üç boyut verisi birçok alanda aktif olarak kullanılmaktadır. Bunlardan en önemlisi doğal bir afet sonrası, bölgede ne tür nesnelerin olduğunu belirleyebilmektir. Herhangi bir dogal afet olmadan da belirli bir bölgede ne tür nesnelerin olduğunu uydu görüntüleri veya üç boyut verisi ile elde etmek şehir bölge planlayıcıları ve devlet kurumları için önemlidir. Türkiyenin yörüngede kendi uzaktan algılama uyduları bulunmaktadır. İleride de gelişmiş ve yeni uzaktan algılama uydularını yörüngeye oturtma planları vardır. Her ne kadar kendi uzaktan algılama uydularımız yörüngede bulunsa da, bunlardan yalnızca ham imge elde edilebilmektedir. Bunlar ya bir operatör aracılığı ile incelenip, içinden bilgi çıkarılabilir; ya da bir yazılım aracılığı ile bilgiler otomatik çıkartılmaya çalışılır. Birinci seçenek her zaman uygulanabilir değildir. Çünkü bu imgelerin boyutu çok büyüktür. Bulunması hedeflenen nesnelerin boyu da imge boyuna göre çok küçüktür. Bu tezde nesne tespiti ve bölütlemesi için yeni yöntemler önerilmiştir. Uzaktan algılama nesneleri, yerel öznitelikler ve şekillerin kombinasyonları olasılıksal oylama çerçevesinde kullanılarak tespit edilmiştir. Nesnelerin şekli uydu görüntüleri ve yükseklik verileri kullanılarak çıkarılır. İlk olarak, uydu görüntülerinde tespit edilen nesnelerin şeklini elde etmek için yeni bir geri yansıtma yöntemi geliştirdik. Ardından yükseklik verileri için iki yeni bölütleme ve filtreleme yöntemi önerilmiştir. İlk yöntem yeni bir morfolojik tabanlı bölge büyütme algoritması ile olasılıklı oylama yöntemine bağlıdır. İkinci yöntem DSM'yi zemin ve zemin olmayan noktalara süzmek ve bölmek için ampirik mod ayrıştırma (EMD) algoritmasını kullanmaktadır. Önerilen yöntemler, farklı uydu görüntüleri (IKONOS, WorldView, QuickBird,) ve üç boyutlu veriler (DSM, LIDAR) üzerinde test edilmiştir. Literatürdeki yöntemlerle karşılaştırıldığında daha iyi sonuçlar elde edilmiştir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES vii
LIST OF TABLES viii
LIST OF SYMBOLS/ABBREVIATIONS ix
1. INTRODUCTION 1
2. LITERATURE REVIEW 4
2.1. Feature Extraction Methods 4
2.2. Object Detection Methods 5
2.3. Contributions 10
3. FEATURE EXTRACTION 13
3.1. Fundamental Feature Extraction Methods 13
3.1.1. Pixel Orientation 13
3.1.2. Directional Edges 14
3.1.3. Steerable Filters 17
3.1.4. Corner Detection 19
3.1.5. Shadow Detection 21
3.2. Feature Vector Formation 22
3.2.1. Edge-Based Ribbons 22
3.2.2. Edge Based L-Shapes 24
3.2.3. Steerable Filter Based Ribbons 27
3.2.4. Steerable Filter Based L-shapes 29
3.2.5. Corner Features 30
3.2.6. Shadow Features 31
3.2.7. Height Features 32
4. PROBABILISTIC OBJECT DETECTION 34
4.1. General Framework 34
4.2. Detection of Simple Shaped Objects 37
4.2.1. Detection using Single Feature Set 38

4.2.2. Detection using Multiple Feature Sets 39
4.2.2.1. Building Detection 44
4.2.2.2. Tree and Ship Detection 45
4.3. Detection of Complex Objects 45
4.3.1. Airplane Detection 45
4.3.2. Car Detection 52
4.4. Experiments on Object Detection 59
4.4.1. Building Detection Experiments 60
4.4.1.1. Experiments with 2D Data 60
4.4.1.2. Experiments with 3D Data 63
4.4.2. Tree Detection Experiments 67
4.4.3. Ship Detection Experiments 71
4.4.4. Airplane Detection Experiments 74
4.4.5. Car Detection Experiments 78
5. SEGMENTATION AND SHAPE EXTRACTION 83
5.1. Shape Extraction in 2D Images 83
5.1.1. Shape Extraction of Simple Objects 84
5.1.2. Shape Extraction of Complex Objects 89
5.1.2.1. Airplane Segmentation 89
5.1.2.2. Car Segmentation 93
5.1.3. Experiments on Satellite Images 94
5.2. Segmentation of Height Data with Voting and Morphology 100
5.2.1. Preprocessing 102
5.2.2. Initial Segmentation 104
5.2.3. Morphological Region Growing 107
5.2.4. Final Non-ground Object Segmentation 107
5.2.5. DTM Generation 110
5.2.6. Experiments on 3D Data 110
5.2.6.1. Parameter Settings 111
5.2.6.2. NSF Open Topography LiDAR Data Set 112
5.2.6.3. WorldView Stereo Image based DSM Data Set 117

5.2.6.4. ISPRS Vaihingen Data Set 119
5.3. EMD Based DSM Filtering 120
5.3.1. Empirical Mode Decomposition 123
5.3.2. IMF Extraction 123
5.3.2.1. Extrema Detection 124
5.3.2.2. Envelope Estimation 125
5.3.3. Applying EMD on a Sample DSM Test Data 125
5.3.4. Iterative EMD based Crude Object Detection 127
5.3.5. Combining Object Detection Results 128
5.3.6. DTM Generation and Fine Object Detection 130
5.3.6.1. DTM Generation 130
5.3.6.2. Fine Object Detection 130
5.3.7. Pseudo Code of the Proposed Method 132
5.3.8. Experiments on 3D Data 133
5.3.8.1. ISPRS LiDAR Data Set 134
5.3.8.2. NSF Open Topography LiDAR Data Set 138
6. CONCLUSIONS 148
REFERENCES 150

LIST OF FIGURES

Figure 3.1. Rectangle and circle shapes with their edges and gradient direction. 15

Figure 3.2. House image and the edges with their gradient direction. 16

Figure 3.3. Directional edges for $\theta = 45^o$ and $\alpha = 30^o$. 17

Figure 3.4. Extracted feature sets. 22

Figure 3.5. The search space for extracting directional edge based ribbon pairs. 23

Figure 3.6. Extracted point ribbon centers for house image for $\theta_n = 60^o$ and $\alpha = 30^o$. 24

Figure 3.7. Two perpendicular ribbons for rectangle modeling. 25

Figure 3.8. L-shape feature extraction steps with directional edges. 25

Figure 3.9. L-shapes with directional edges and steerable filters. 26

Figure 3.10. Steerable filtering result for $\theta = 0^o$, $\theta = 45^o$, and $\theta = 90^o$ respectively. 28

Figure 3.11. Steerable filter for L-shaped rectangle model. 29

Figure 3.12. Corners for rectangle model. 30

Figure 3.13. Shadow feature vectors illustration. 31

Figure 4.1. Desired pdf estimation of object center locations. 35

Figure 4.2. Proposed high level framework for object detection from satellite images. 36

Figure 4.3. Simple and complex shape object samples from satellite images. 38

Figure 4.4. Noisy and low resolution panchromatic images of buildings.................. 40

Figure 4.5. Detecting rectangular shapes using multiple features............................ 41

Figure 4.6. Detecting ellipsoidal shapes using multiple features............................. 41

Figure 4.7. Multiple rectangles model for airplanes. 46

Figure 4.8. Sample satellite image of airplane and Canny edges............................. 47

Figure 4.9. metry of airplane parts and possible voting locations............................ 48

Figure 4.10. Voting method for airplane parts.. 49

Figure 4.11. Airplane center pdf estimates for $\theta_i = 30^o$... 50

Figure 4.12. Multiple rectangles model for cars. 52

Figure 4.13. Sample satellite image of car and Canny edges.................................... 53

Figure 4.14. The relative geometry between car parts and voting locations. 54

Figure 4.15. Edges and rectangle centers for $\theta = 0$... 55

Figure 4.16. Vote locations for $\theta = 0^o$... 56

Figure 4.17. The weight function for voting car center. 57

Figure 4.18. Car center pdf estimates for $\theta_i = 0^0$ 58

Figure 4.19. Samples for Adana satellite images. 61

Figure 4.20. Estimated pdf of buildings for each feature for Adana7 test image. 61

Figure 4.21. Final estimated pdf using all five features and building detection results. .. 62

Figure 4.22. PR curve for 23 Adana test images with 674 buildings in total. 64

Figure 4.23. Test images of Vaihingen, Germany. 65

Figure 4.24. Estimated pdf of buildings for each future for $top\_mosaic\_09cm\_area3$. .. 65

Figure 4.25. Final estimated pdf using all five features and building detection results. .. 66

Figure 4.26. PR curve for 8 ISPRS Vaihingen images with 323 buildings. 67

Figure 4.27. Test images used in tree detection experiments. 68

Figure 4.28. Satellite images of ships. 72

Figure 4.29. Sample image and edges from steerable filter result. 73

Figure 4.30. Estimated pdf of ship centers and ship detections. 73

Figure 4.31. Airplane images: Top row Image 1-3-5, bottom row Image 7-9-11. 75

Figure 4.32. Detection results for Image 11 76

Figure 4.33. FP examples for airplane detection. 77

Figure 4.34. PR curve for 12 airport test images with 170 airplanes in total. 79

Figure 4.35. Car detection results on a subset of ISPRS Vaihingen patch.................... 79

Figure 4.36. Pdf estimations of car centers.. 80

Figure 4.37. PR curve for ISPRS Vaihingen test images with 187 cars in total............ 82

Figure 5.1. Vote back-projecting method for object outline extraction...................... 84

Figure 5.2. Steps of the vote back-projecting method for object outline extraction. 85

Figure 5.3. Sample satellite image of buildings, Canny edges and pdf estimation....... 85

Figure 5.4. Back-projection and segmentation steps.. 86

Figure 5.5. Steps of the vote back-projecting method for airplane shape extraction..... 90

Figure 5.6. Sample satellite image of airplanes, Canny edges and pdf estimation. 91

Figure 5.7. Segmentation results for the image of airplanes.................................. 92

Figure 5.8. Steps of the vote back-projecting method for object outline extraction. 93

Figure 5.9. Segmentation results for cars.. 93

Figure 5.10. Segmentation results on some of the Ikonos satellite images of Adana. 95

Figure 5.11. Segmentation results on some of the satellite images of orchard trees....... 97

Figure 5.12. Segmentation results on some of the satellite images of ships. 98

Figure 5.13. Segmentation results on some of the satellite images of airports............. 99

Figure 5.14. Segmentation results of cars on ISPRS Vaihingen data set..................... 101

Figure 5.15. The original Utah-5 DSM test data without preprocessing..................... 103

Figure 5.16. Demonstration of the preprocessing and small object detection steps. 104

Figure 5.17. Initial segmentation results on non-ground and ground. 105

Figure 5.18. Mmorphological region growing algorithm on the initial building segment.108

Figure 5.19. Morphological region growing method on a ground segment................. 109

Figure 5.20. Non-ground object detection results for the Utah-5 DSM test data.......... 109

Figure 5.21. DTM for the Utah-5 DSM test data. 110

Figure 5.22. Utah DSMs. 1 row: Utah 1-3; second row: Utah 4-6; third row: Utah 7-9. 112

Figure 5.23. Generated DTMs for Utah-1, Utah-2, Utah-7, Utah-8 and Utah-9 test data.113

Figure 5.24. Detailed object detection results for the Utah 1-2-7-8-9 test data. 114

Figure 5.25. Filtering results on the WorldView stereo image based DSM data set....... 118

Figure 5.26. Building detection results on the Vaihingen 1-3-5 of ISPRS data set. 121

Figure 5.27. Flowchart of the proposed EMD based DSM filtering method. 122

Figure 5.28. The Utah3 DSM test data. 126

Figure 5.29. Vertical profile of the Utah3 DSM test data with line number 650........... 126

Figure 5.30. Residues obtained from the vertical profile of the Utah3 test data. 127

Figure 5.31. Crude object detection results from the first to the fourth residue............ 129

Figure 5.32. Combined crude object detection result for the Utah3 test data............... 130

Figure 5.33. DTM generation result on the Utah3 test data. 131

Figure 5.34. Fine object detection results on the Utah3 test data. 132

Figure 5.35. Visual DTM generation results for some ISPRS data samples. 136

Figure 5.36. Visual object detection results for some ISPRS data samples. 137

Figure 5.37. The effect of parameters on the performance for the ISPRS data set......... 139

Figure 5.38. DTM generation results on forest region test samples. 141

Figure 5.39. Object detection performances in terms of Kappa scores on forest regions.142

Figure 5.40. DTM generation results on residential region test samples. 143

Figure 5.41. Object detection results on Utah test samples. 144

Figure 5.42. Kappa scores of detection results on residential regions. 145

Figure 5.43. Interval number (N) versus the computation time for the ISPRS data set... 146

LIST OF TABLES

Table 4.1. Building detection results for Adana images. 63

Table 4.2. Building detection results for ISPRS Vaihingen data set.......................... 66

Table 4.3. Metadata for test images of orchard trees. 69

Table 4.4. TP, FP, and FN values using different edge detectors. 69

Table 4.5. CP, CR, and Q values in percentages using different edge detectors. 70

Table 4.6. Comparison of the proposed method with Ozdarici-Ok's method. 71

Table 4.7. Ship detection results. 74

Table 4.8. Metadata for test images of airplanes. 75

Table 4.9. Airplane detection performance for different detection thresholds. 78

Table 4.10. Proposed method's performance on the ISPRS Vaihingen data set. 81

Table 5.1. Performance results on the NSF Open Topography LiDAR data set. 116

Table 5.2. Performance results on the WorldView DSM data set. 119

Table 5.3. Performance on the ISPRS Vaihingen, Germany data set. 120

Table 5.4. Comparison with five other methods in literature on the ISPRS data set...... 135

Table 5.5. Metadata for the NSF Open Topography LiDAR data set. 140

Table 5.6. Computation times in seconds on the NSF Open Topography data set. 147

LIST OF SYMBOLS/ABBREVIATIONS

CM	Completeness
CR	Correctness
CS	Convex Shape
DSM	Digital Surface Model
DTM	Digital Terrain Model
EMD	Empirical Mode Decomposition
FAST	Features from Accelerated Segment Test
FP	False Positive
FN	False Negative
GHT	Generalized Hough Transform
GIS	Geographic Information System
HOG	Histogram of Oriented Gradients
IMF	Intrinsic Mode Functions
ISPRS	International Society for Photogrammetry and Remote Sensing
LiDAR	Light Detecting and Ranging
NDVI	Normalized Difference Vegetation Index
NSF	National Science Foundation
PDF	Probability Density Function
PR	Precision and Recall
RGB	Red, Green and Blue
SAR	Synthetic Aperture Radar
SIFT	Scale-invariant Feature Transform
SIFT	Speeded Up Robust Features
TIN	Triangulated Irregular Network
TOP	True Orthophoto Image
TP	True Positive
UAV	Unmanned Aerial Vehicle

1. INTRODUCTION

Information retrieval from remotely sensed data has gained wide application areas due to increasing number of aerial and space sensors and sensor resolution. Also accessing the remote sensing data has become easier with commercial satellite sensors which push authorities measuring any information from Earth surface for various reasons. Remote sensing applications mainly interested in detection and classification of objects on Earth. These include detecting buildings for urban planning, any natural disaster organization for emergency responses, locating hidden urban locations or planning military operations for countless reasons. Remote sensing applications also interested in detecting other objects such as cars for monitoring vehicle traffic flow and intelligent transportation planning in cities, trees in orchards to manage and forecast crop yield, ships for fishery management, maritime traffic monitoring, and security etc.

For these purposes, mostly optical satellite images are used. However, the available data is increasing such as multispectral data with various frequency bands and DSM (Digital Surface Model) or LiDAR (Light Detecting and Ranging) data for height information. Not only the data type, but also the spatial resolution of the data is also increasing. This led people to investigate newer methodologies to analyze the finer details of objects which were not even seen with bare eyes in low spatial resolution images before. Due to the large data sizes, it is almost impossible to analyze the data manually. So researchers propose automated methods to reduce the complexity of the problem. For this purpose, computer vision methods are heavily used on remote sensing data.

In computer vision applications, segmentation is considered as a crucial step in the image processing chain. Segmentation helps reducing the complexity of the problem and thus helps understanding the environment pictured in the sensory data. Perceptual organization of the features in the sensory data is a way of segmentation of the data where the features are grouped and organized to help to see the whole meaning in the data.

It is believed that our world is not visually chaotic [1]. So organizing the features in sensory data should help understanding the scene in the image. Actually, grouping and structuring the features has been a study for Gestalt psychologists. The Gestalt principles tell the importance of the organization in human vision system. They showed that the most probable perceived meaning in human vision system is the one with most regular and stable one which is the one furthest from randomness. Thus, it is believed that the human vision system looks for the spatially organized features in the image.

According to Gestalt principles, there are some basic properties in real world images that helps our vision system organizing features. Some of these properties are symmetry, continuity, proximity and closure [2]. Using these properties, it is possible to reduce the data for higher level cognitive processes. In this thesis, a generic framework is proposed to detect simple and complex shapes in images using the basic Gestalt principles. The proposed framework is applied for object detection in remote sensing data such as satellite images and height data. It is shown that with simple assumptions on object shapes and features, it is possible to detect and segment objects in a complex environment with a probabilistic framework.

Many object detection techniques utilize machine learning algorithms where the algorithm itself needs training data. Preparing the training data is extremely painful. Besides, it is almost impossible to handle all cases in the training data set. In this thesis, an unsupervised method is proposed for object detection in remote sensing. The outcome of the proposed framework is separation of objects and background in an image. While separating objects from the background, object types and their shapes are also extracted.

It is assumed that we have a sensory data output which might include panchromatic images, RGB images, multispectral images, DSM images, etc. In the first step, some feature extraction methods are applied on the available data such as corner detection, edge detection, etc. Then these features are grouped into meaningful parts with a probabilistic voting framework. At this step, it is assumed that objects in the images have simple or combination of simple shapes. Thus, the probabilistic voting framework tries to find the center locations of these shapes. At the last step, the outlines of the objects are extracted.

This thesis is divided into six chapters. Chapter one introduces the overview of the thesis and reviews the contributions. We start with relevant literature review in the second chapter. The feature extraction methods are given in chapter three. In chapter four, the proposed probabilistic representation and object detection methods are explained in detail. Also the experiments for object detection in remote sensing data is given at the end of the chapter. In chapter five, shape extraction and segmentation methods are given. The experiments of the proposed methods are also given at the end of the chapter. The results are compared with other methods in the literature. Finally, in chapter six the proposed methods are concluded with suggestions of further developments.

2. LITERATURE REVIEW

Computer vision applications are attracting more attention in the recent years due to the increase in technological developments. The researchers are also paying more attention in these applications. So new methods emerge almost every day. We heavily utilized low level features in images and then detected and segmented objects in remote sensing images. In this chapter, we review the related feature extraction and object detection methods in the literature.

2.1. FEATURE EXTRACTION METHODS

Even though new methods emerge almost everyday, the fundamentals of image processing applications lean on fundamental feature extraction methods from an image. Low level feature extraction methods find interest points in the image. Then these are used in many applications such as detection, shape matching, tracking, recognition etc. The most known and used low level feature is edges which are extracted point based. Edges are extracted mostly with first and second order differential operators. The most popular edge detecting methods are Sobel operator [3], Canny [4] and Marr-Hildreth [5]. The extracted edges and their orientation are heavily used in computer vision applications. Other mostly used features are corners. Among many methods, Moravec corner detection is one of the earliest [6]. Harris corner detector [7], which we also use in this dissertation, is a modified version of Moravec's. FAST [8] is one of the high-speed corner detection methods which tests if a portion of the neighbor pixels brightness is higher or lower than then the reference pixel's brightness plus a threshold. In [9] and [10] machine learning based corner detection algorithms proposed which need less processing power compared to other methods and thus more suitable for real time scenarios. After detecting corners, or other interest points, feature descriptors are defined for each feature which describes the patches around these points. These two together, interest point and descriptor, are generally called local features. SIFT [11] and SURF [12] are robust, scale and rotation invariant descriptors that are used mostly in applications which require finding matching points of two images.

2.2. OBJECT DETECTION METHODS

In this thesis, we are interested in detection of objects in remote sensing images. First we give brief information on some of the known object detection methods for ground-shot images. Then, we will give a review of the related field.

Generalized Hough Transform (GHT) [13] is a method for finding arbitrary shapes in grey level images. In this method, the boundaries of an object are mapped to a Hough transform space in which the edges of the object are stored in a table with orientations wrt a reference point. Then, this table is used to detect arbitrary shapes in an image. Implicit shape model [14] method similarly uses a codebook of appearances of the different parts of an object. Beforehand, each part of the object is trained for the object center voting process. So if a patch around an interest point finds a match in the codebook, it votes for the possible object center. The object detection and segmentation is done with a probabilistic framework. Class specific Hough forests [15] also use GHT where for the detection of object parts and object center voting, random forests are used. These methods need training beforehand and mostly used for objects that are deformable in parts such as cars, pedestrians, animals etc.

For the remote sensing applications we first focus on building detection methods. Detecting and locating buildings in satellite images has various application areas. Unfortunately, manually detecting buildings is hard and very time consuming. Therefore, in the literature several methods are proposed to automatically detect buildings. In general, the researchers developed algorithms according to the data type available. Researchers used panchromatic, multispectral or height information to detect buildings. Multispectral information are mostly used in detection of non-building areas such as shadow [16–19], vegetation and water [20–22]. So that the search space for buildings is reduced. In [23], local feature extraction methods are utilized for detection of possible building shapes such as rectangulars. They used corners, FAST and SIFT features in a probabilistic voting framework to detect buildings in panchroamtic images. There are also supervised learning methods [24–27] in which with the ground truth data available, classifiers are utilized in pixel or segment level. There are several works using DSM for building detection and 3D reconstruction. Most of them use the height information to remove non-building structures and focus on the building shape and rooftop contours. Tournaire *et al* [28] used point processes on digital elevation models. They calculated an energy function for fitting rectangles on buildings based on the adequacy of objects and prior knowledge to extract footprint of buildings. Ortner *et al* [29] used two interacting spatial point processes on DEM to fit rectangular shapes on building segments. Brunn and Weidner [30] separated buildings and vegetation areas using height and geometric information on DSM data. After detecting buildings, they used surface normals to extract rooftop geometries. Sirmacek *et al* [31] used DSM for detecting building ground floor shapes using an active shape detection approach. Then, they used derivative filters to extract roof ridge lines. This leads to 3D building reconstruction. Galvanin and Poz [32] proposed a method for rooftop extraction. They used DSM data to detect above ground objects. Therefore, they segmented DSM with a recursive splitting technique and region merging process. Awrangjeb *et al* [33] also proposed a method to separate buildings and trees using DSM. They used height and width information from DSM with a ground mask. They used the image entropy and color information to remove trees. Most previous works assume that thresholding DSM provides sufficient information about the building shape. Unfortunately, using local thresholding for DSM data fails at industrial areas where big buildings are closely located. In these areas, the window size for local thresholding needs to be very large. Also due to automatic DSM generation, some unwanted outliers may occur. These are caused by matching errors, temporal changes or applied interpolation techniques. These also affect the building detection process in the negative manner. As an example, closely located buildings in city areas cause uncertainty on building edges. The main reason for this is the applied interpolation technique which causes a loss of sharpness. Buildings also do not have clear rooftop contours because of the mentioned reasons. Sometimes a group of trees may look like a building and there is no easy way to separate them. However, the height information in DSM is still very valuable.

In the second category, we give a review for tree detection methods. Detecting and delineating tree crowns in satellite images is important subset of object detection problem in remote sensing. Most of the general object detection methods rely on training based classifiers. In this and most other tree crown detection methods, a classifier is not used. Instead, specific clues for trees are used. Tree crown detection methods can be grouped into four categories as local maxima filtering, image thresholding, scale analysis, and template matching. The first two methods are used extensively in literature [34, 35]. Since tree tops reflect the light falling on them, they will be seen as bright spots in satellite or aerial images. The reflection decreases from top to bottom of a tree. Therefore, bottom parts of the tree will be seen darker. Local maxima filtering method uses this information [36–39]. In thresholding based methods, bright and dark regions are obtained in the image. To do so, well known image thresholding methods are used [40, 41]. It is observed that spatial resolution of the image and size of a tree is important for detection. If the spatial resolution (more specifically the ground sampled distance) of the pixel is smaller than the tree, then the user should look at the image from different scales. Then, these are used in tree detection. This method is also used to detect different sized trees [42–46]. A simple tree template is used to detect trees in template matching methods. However, this may not be feasible since trees in neighboring orchards in a single satellite image need not be uniform in terms of size, shape, and height. We can group tree crown delineation (boundary extraction) methods into three categories as: valley following, region growing, and watershed segmentation. In the valley following method, tree boundaries are extracted based on shadow between them [47–50]. In region growing, tree tops are generally selected as the starting point. Then, the region grows by inspecting neighboring points. Based on a predefined criterion, the growing stops. This way, tree boundaries are segmented [37, 51–53]. In watershed segmentation, first the negative of the grayscale image is obtained. Then, local minima is obtained using watershed segmentation. This region corresponds to the tree crown [38, 52]. Shape information, classification based on several features, and texture information can also be used to detect and delineate tree crowns [54–57]. Recently, [58] proposed a method for citrus tree crown detection and delineation using fast radial symmetry transformation. This paper provides an excellent literature review on tree crown delineation. [59] also provide an excellent review on tree crown detection and delineation methods till 2011. [60] also offer a recent review on the same subject. Although the mentioned methods work fairly well in detecting trees, there are several problems mentioned in literature. First, multispectral information is used in some studies which may not be available in operation. Second, trees in neighboring orchards (present in one satellite or aerial image) may not be uniform in terms of size, shape, and type. Therefore, it may not be possible to handle these variations by a single method. Third, the background color in the orchard may not be the same for all test images. Fourth, shadow data may not be useful in the image due to the time of the day image is taken. Hence, methods based on this information may not work properly. Fifth, there may be man-made objects in the image besides trees. This may further increase the complexity of tree crown detection and delineation problem.

In the third category we review ship detection methods in remote sensing. Detecting and locating ships in satellite images can be used for several purposes. The most promising and emphasized of these can be counted as fishery management, maritime traffic monitoring, and security. In all these applications, a selected region should be monitored to detect and locate possible ships. To solve this problem, researchers proposed several methods using SAR images. Although these images have several advantages, they also have disadvantages. Therefore, new methods have emerged based optical satellite images. Recent work on optical satellite image based ship detection can be summarized as follows. Zhu *et al.* [61] proposed a hierarchical method based on shape and texture features. Here, the sea regions are assumed to be detected beforehand. This study also summarizes the advantages and disadvantages of SAR based ship detection methods in detail. Proia and Page [62] proposed a ship detection method on optical satellite images using Bayesian decision theory. Corbane *et al.* [63] proposed a method to detect ships. This method consists of preprocessing, mathematical morphology, connected components analysis, logistic regression, wavelet and Radon transform steps. Bi *et al.* [64] proposed a method based on visual attention mechanism. This method works on a hierarchical manner with a multiscale approach. Yang *et al.* [65] approached the ship detection problem from a different perspective. They focused on sea surface analysis. Then, they detected ships using texture and shape information. All these studies used SPOT 5 satellite images. Shi *et al.* [66] proposed a method based on HOG features and the Adaboost method. They also grouped previous ship detection methods into three categories as thresholding, statistical gray value distribution, and classification based. Xu *et al.* [67] focused on inshore ship detection. They proposed a method based on rotation invariant generalized Hough transform. Liu *et al.* [68] also focused on inshore ship detection. They proposed a method based on active contour models and shape analysis. These studies benefit from Google Earth images for ship detection. Tang *et al.* [69] recently proposed a ship detection method using wavelet coefficients, deep neural networks, and extreme learning machines.

Next we give methods in the literature for car and airplane detection. There are several car detection methods. Yao *et al.* [70] used airborne laser scanning (ALS) data for vehicle detection. They first apply ground level separation to extract the region of interest. Then, they applied marker controlled watershed segmentation with morphological reconstruction to detect vehicles. Moranduzzo and Melgani [71] used UAV images to count cars. This method first extracts asphalt zones. It then extracts features from these regions. Afterwards, SVM is used for classification. Toth and Grejner-Brzezinska [72] used ALS data to monitor traffic flow. They used GIS data for road mask and bounding box estimation along road surface. There are also related work on other sensors such as SAR and infrared cameras [73–76]. Also, there is a good review on vehicle detection in high resolution satellite images [77]. In airplane detection methods, sliding window and supervised learning based approaches are used. Cheng *et al.* [78] used histogram of gradient (HOG) features in a discriminatively trained mixture model. Han *et al.* [79] merged visual saliency modeling and the discriminative learning of sparse coding for various object detection including airplanes. Hough forest methods mainly focus on detecting objects with single orientations only. Lei *et al.* [80] proposed using a color enhanced rotation invariant Hough forests for detecting remote sensing objects. Yu *et al.* [81] and Qiu *et al.* [82] also used Hough forest for airplane detection in remote sensing images. All of these methods require a training step.

In the final step, we review the LiDAR and DSM filtering methods. Sithole and Vosselman [83] grouped DSM filtering methods into three categories as slope, linear prediction, and mathematical morphology based. Based on the recent work in this area, three more categories can be added as segmentation, statistical, and neural network based. Slope based methods assume that the elevation change between an object and the neighboring ground point will be abrupt [68, 83–86]. Methods based on this approach are successful on flat regions. However, their performance decreases at hilly regions. Linear prediction based methods generate an approximation of bare Earth's surface and threshold objects above ground [87–91]. In linear prediction, interpolation methods are used most of the time. These may misclassify small objects. There are also newer methods in this category. Mongus and Zalik [92] proposed a parameter free method for DTM generation. Chen *et al.* [93] improved this method further. They proposed a multiresolution hierarchical classification algorithm based on LiDAR point residuals from the interpolated raster surface. Zhang and Lin [94] proposed an updated progressive TIN densification method using point cloud segmentation. Mongus and Zalik [95] proposed a multi-scale decomposition method which uses connected operators. The advantage of this method is its computational efficiency. Hu *et al.* [96] recently proposed an adaptive surface filtering method using regularization. They obtained very good results in detecting objects on a standard airborne LiDAR test data. Morphology based methods are based on dilation and erosion operations to extract ground points [97–99]. Methods using this approach can adopt to various region types. The problem in morphology based methods is finding an appropriate structuring element for different region types. Recently, there have been improved methods in morphology based category. Pingel *et al.* [100] proposed a method which iteratively applies morphological opening with increasing window size at each iteration. This method uses image inpainting to generate the DTM. Mongus *et al.* [101] proposed a method which uses differential morphological profiles to form a tophat scale-space. They used this method to extract buildings from LiDAR data. Li [66] proposed a morphological filtering algorithm based on multi-gradient analysis. Segmentation based methods are mostly used for land cover classification using LiDAR point cloud data [102–104]. These methods try to segment the data with heuristic features. Statistical methods assume that LiDAR ground points have a normal distribution. On the other hand, LiDAR object points tend to break the symmetry of the normal distribution, thus causing skewness [105–108]. Therefore, these methods try to remove object points until the skewness is balanced. Jahromi *et al.* [109] recently proposed a different method based on artificial neural networks to extract bare-earth points from airborne laser scanning data. They obtained results for both semi-automatic and supervised training data.

2.3. CONTRIBUTIONS

In this thesis, the detection of objects is realized with a probabilistic voting scheme. There are probabilistic voting methods for object detection such as implicit shape modeling, Hough forests and generalized Hough transform. In these methods, the parts of an object use votes for detecting the center of the object. To do this, the parts are trained to vote for the correct location.

In the proposed framework, there are very basic differences. Here we differ from the implicit shape modeling and Hough forest methods with the training section. We do not use training sets for learning where to vote for each feature. In the proposed approach, the training is implicit within the shape model. It is assumed that the objects have either simple shapes or combination of simple shapes. For each object, specific rules are defined and it is enforced that the extracted features with these rules vote for object centers. Actually when the certain shapes of some objects are considered in satellite images such as house, it is difficult to train the features for all cases of house images. Other training based methods try to separate parts of a specific object and train the parts to find the object. However in the house image case, where it is simply a rectangular shape, it is difficult to define individual parts for training. Indeed we use general observations on simple shapes and use them for probabilistic voting.

Generalized Hough transform finds the parameters of the system, where we describe the object and search for the object centers directly in the spatial space. Thus we do not try to find any parameters for the object. The first contribution of the thesis is utilizing simple features for a rule based object center detection in a probabilistic manner. In the probabilistic voting approach, vote maps are generated for each feature type. The combination of voting maps are realized with the Bayes's theorem. Bayes says that the initial probability of an hypothesis might be updated based on the related evidences. Basically, for each additional feature, the probability map of object centers is updated.

After detection of object centers, the shape of the objects are extracted. The rest of the contributions belong to segmentation and shape extraction. The second contribution of the thesis is an algorithm for extracting shapes of objects. Here we propose an algorithm for extracting the outline of the object using a back-projection algorithm. Here we did experiments on panchromatic images and obtained good results.

The third and fourth contribution are segmentation algorithms of height data into ground and non-ground pixels. In the first algorithm, a novel ground filtering and region growing segmentation method is proposed for DSM data. The proposed method uses the local maxima of probabilistic voting map where the modes of map are used as seed points in segmentation. Here, a novel segmentation method is proposed based on morphological thicken operations. The experiments show that the proposed algorithm has certain advantages for filtering large size objects compared to other algorithms in the literature.

The fourth and last contribution is filtering and segmenting DSM data with EMD algorithm. Local, nonlinear, and non-stationary characteristics of EMD allow better DTM generation and object filtering. The proposed method is tested on two publicly available LiDAR data set and promising results are obtained. Besides, the proposed method is compared with other methods in the literature. Comparison results indicate that the proposed method has certain advantages in terms of performance.

3. FEATURE EXTRACTION

In this chapter, some of the well-known local and structural feature extraction methods in image processing are revised such as Harris corner detector [7], steerable filters [110] and Canny edge detector [4]. In addition to these, we developed new features for object detection in remote sensing images. These are based on gradient orientation of extracted edges in a given image.

There are lots of data to be processed in a given image. To reduce the amount of data and find evidence of objects in the image, we find interest points with feature extraction methods. The extracted features are either in pixel level such as isolated points, in structural level or connected region level. We investigate subsets of all three categories. Once we extract features in the image, we extract complimentary attributes such as edge orientation and gradient magnitude. The extracted features will be used in a probabilistic framework for object center location detection in the next chapter. In this chapter, only the feature extraction methods are considered.

3.1. FUNDAMENTAL FEATURE EXTRACTION METHODS

There are several well-established feature extraction methods in literature. We review the ones which will be used in the following chapters in this section.

3.1.1. Pixel Orientation

In computer vision applications, pixel orientation is heavily used to extract information from images performed by gradient operators. In image analysis, the gradient of image means the derivatives of image intensity for each pixel at horizontal, x, and vertical, y, directions. In a small neighborhood, the gradient direction indicates the largest possible intensity increase direction. Since image is a two dimensional discrete data, it is only possible to define derivatives with simple assumptions such as image has a continuous intensity function and sampled at pixel locations.

Generally the image is convolved with a kernel and gradient operation is performed. Therefore, we convolve the image with a Gaussian smoothing kernel and find x and y gradients as follows.

$$I_x(x, y) = I(x, y) * \frac{-x}{2\pi\tau_g^4} \exp(-\frac{x^2 + y^2}{2\tau_g^2}) \tag{3.1}$$

$$I_y(x, y) = I(x, y) * \frac{-y}{2\pi\tau_g^4} \exp(-\frac{x^2 + y^2}{2\tau_g^2}) \tag{3.2}$$

where $*$ stands for the two dimensional convolution operation. Here, τ_g is the smoothing parameter. As we obtain I_x and I_y, we calculate the gradient direction using

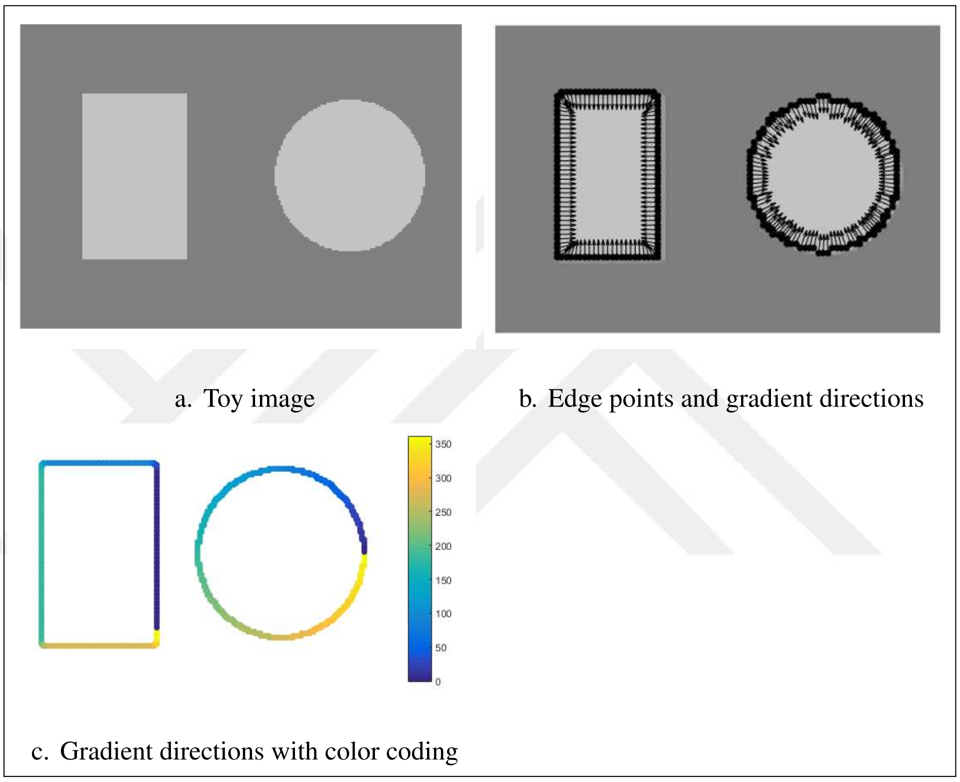
$$O(x, y) = \arctan\left(\frac{I_y(x, y)}{I_x(x, y)}\right) \tag{3.3}$$

For a pixel with coordinate (x_i, y_i), the corresponding gradient direction is $\theta_i = O(x_i, y_i)$.

3.1.2. Directional Edges

Edges are strong indicators of existence of an object. Therefore, it is common to use edge detector results at structural level with connected component analysis. However, since each edge pixel location calculated by the edge detector is a possible object boundary, we utilize the edge detection results in pixel level where each edge point is handled individually. We pick Canny edge detector due to its scalability and robustness to noise [4]. Suppose the extracted edge points are represented as (x_i, y_i) for $i = 1, 2, \cdot\cdot\cdot, N$ where N is the number of edge points. We call the edge detector result as $B(x, y)$ where it has binary values as one for edge pixels and zero for non edge pixels.

Consider the toy image in Figure 3.1.a. The extracted edge points and their gradient direction are given in Figure 3.1.b. We also plot edges in different colors according to their gradient directions in Figure 3.1.c. If the object color is brighter than background, then the edges are showing the center of the object.



a. Toy image

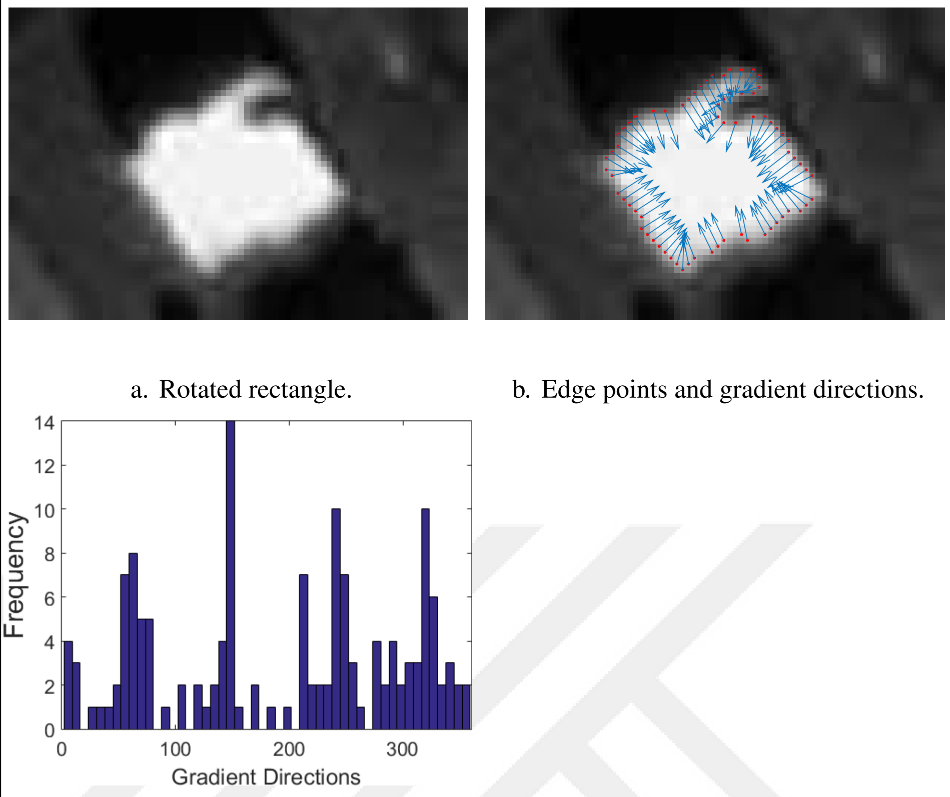
b. Edge points and gradient directions

c. Gradient directions with color coding

Figure 3.1. Rectangle and circle shapes with their edges and gradient direction.

In the following sections, it will be shown that location of the edge point and its orientation can be used to setup a vote for the possible object center location. This is done by pairing two edge points where they have opposite directions such as $\theta_i = 180 + \theta_j$. Unfortunately, in real images it is difficult to find two points that have exactly opposite directions. We give a satellite image of a single house in Figure 3.2. As can be seen here, the house does not have a perfect rectangular shape. Besides, the edge gradient directions are not separated with 90^o which is clearly seen in the histogram in Figure 3.2.c.

Instead of using a single direction, we group edge points with respect to their gradient



a. Rotated rectangle.

b. Edge points and gradient directions.

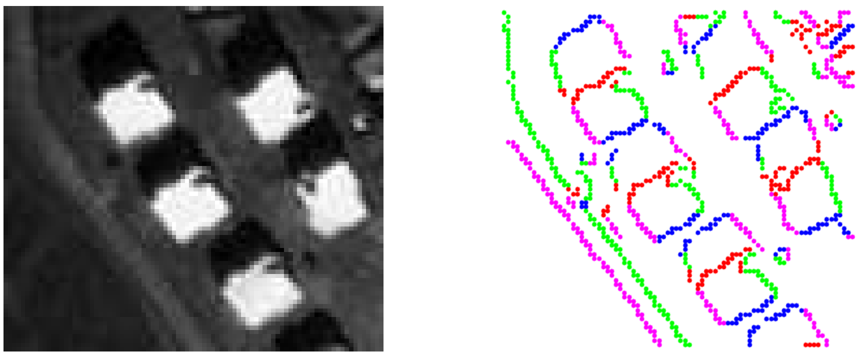
c. Histogram of the gradient directions.

Figure 3.2. House image and the edges with their gradient direction.

directions. Let's analyze this approach further on the house example. First, we define a direction interval such that for a specific direction θ we define a direction interval $\beta[\theta, \alpha] = [\theta - \alpha, \theta + \alpha]$ with

$$\beta[\theta, \alpha] = \{x \mid \theta - \alpha \leqslant x \leqslant \theta + \alpha\} \tag{3.4}$$

where α is the half interval length. Now, we can group edge points that are in β direction. Extracted edge points with 90^o apart are given in Figure 3.3. The edge points which have the direction interval β is represented as $e_{\beta i} = (x_j, y_j)$ for $j = 1, 2, \cdots, J$ where J is the number of edge points in the specified gradient direction. Then, we define angle intervals with 90^o



a. Grayscale satellite image of houses. b. β'

Figure 3.3. Directional edges for $\theta = 45^o$ and $\alpha = 30^o$.

apart with respect to the first angle interval as

$$\beta' = \beta + 90^o$$

$$\beta'' = \beta + 180^o \tag{3.5}$$

$$\beta''' = \beta + 270^o$$

In Figure 3.3.b, we give extracted edges for $\theta = 45^o$ and $\alpha = 30^o$ where e_β in magenta, $e_{\beta'}$ in red, $e_{\beta''}$ in green and $e_{\beta'''}$ in blue color dot markers. Thus, the four edges of the rectangular shapes are extracted individually. Also, on the left side of the image, edges of the road are separated side by side.

3.1.3. Steerable Filters

Edges are crucial features to detect objects in remotely sensed images. However, the objects are located in various angles where for some cases a simple edge detector won't work to detect all edges successfully. For such cases, we can benefit from the steerable filters [110].

Steerable filters can be applied in different orientations. Hence, directional edge-like structures can be extracted. Steerable filters are synthesized easily as a linear combination of a set of basis filters. For a symmetric Gaussian function, the basis filters $G_1^{0^0}$ and $G_1^{\pi/2}$ are

$$G_1^0 = \frac{\partial}{\partial x} e^{-(x^2+y^2)} = -2xe^{-(x^2+y^2)} \tag{3.6}$$

$$G_1^{\pi/2} = \frac{\partial}{\partial y} e^{-(x^2+y^2)} = -2ye^{-(x^2+y^2)} \tag{3.7}$$

For an arbitrary orientation θ, the filter G_1^θ can be synthesized by taking a linear combination of the basis filters as

$$G_1^\theta = cos(\theta)G_1^0 + sin(\theta)G_1^{\pi/2} \tag{3.8}$$

When the image $I(x, y)$ is convolved with the steerable filter function in θ direction as

$$J^\theta(x, y) = G_1^\theta * I(x, y) \tag{3.9}$$

then we will have a high response if any structure that is perpendicular to θ exists.

Man-made objects mostly have symmetrical shapes such as two parallel edges. When we apply a steerable filter which is perpendicular to the side edges of the object, we will get a high positive response from ground to object passing and high negative response from object to ground passing. Therefore, we will have positive and negative responses at the side edges of the object. Since objects may be in any orientation, we should apply steerable filter at multiple orientations such as for N directions as $\theta \epsilon [0, \pi/N, 2\pi/N, ..., (N-1)\pi/N]$. Thus, the steerable filter response of the image for θ direction will be $J^{\theta}(x, y)$.

3.1.4. Corner Detection

Corners are interest points in the image which are heavily used in many applications such as image matching, 3D modeling, and object recognition. Corner may be defined as the intersection of two edges. Corner detectors are basically detecting the gradient direction change in the image. In satellite or aerial images when an object exist in the image, we see several corner points. These are evidence of object existence in the image.

Even though there are other corner detection algorithms, we preferred Harris corner detector in this study [7]. This detector has invariance to rotation, scale, illumination variation, and image noise. It uses an auto-correlation function and measures local changes in the image in a small neighborhood of different directions.

Let's call $I(x, y)$ the image where (x, y) denotes the location. The auto-correlation function is defined as

$$E(x, y) = \sum_{W(x,y)} G(x_i, y_i)[I(x_i, y_i) - I(x_i + \Delta x, y_i + \Delta y)]^2 \qquad (3.10)$$

where $(\Delta x, \Delta y)$ is the amount of shift; (x_i, y_i) are point locations in a window $W(x, y)$; and G is a Gaussian kernel function.

Both W and G are centered on (x, y). If $(\Delta x, \Delta y)$ is small, then the shifted image may be rewritten with a truncated Taylor series expansion as

$$I(x_i + \Delta x, y_i + \Delta y) \approx I(x_i, y_i) + \begin{bmatrix} I_x(x_i, y_i) & I_y(x_i, y_i) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{3.11}$$

where I_x and I_y denote the partial derivatives in the x and y directions, respectively. Thus, the autocorrelation function in Eqn. 3.10 becomes

$$\begin{aligned} E(x, y) &= \sum_{W(x,y)} G(x_i, y_i)(\begin{bmatrix} I_x(x_i, y_i) & I_y(x_i, y_i) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix})^2 \\ &= \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} M(x, y) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \end{aligned} \tag{3.12}$$

where $M(x, y)$ is

$$M(x, y) = \begin{bmatrix} \sum_{W(x,y)} G(x_i, y_i) I_x^2(x_i, y_i) & \sum_{W(x,y)} G(x_i, y_i) I_x(x_i, y_i) I_y(x_i, y_i) \\ \sum_{W(x,y)} G(x_i, y_i) I_x(x_i, y_i) I_y(x_i, y_i) & \sum_{W(x,y)} G(x_i, y_i) I_y^2(x_i, y_i) \end{bmatrix} \tag{3.13}$$

Here the matrix $M(x, y)$ is the key point of Harris corner detector. Eigenvalues of $M(x, y)$ give important information of local changes in the image. If both eigenvalues are small, then $E(x, y)$ vanishes. This tells that the small neighborhood around (x, y) has a small change. If only one eigenvalue is high at (x, y), then it is an edge passing. If both eigenvalues are high, then the response of $E(x, y)$ will be a peak at (x, y). This tells that, tells there is a corner there. Instead of calculating the eigenvalues, the response function below is computed.

$$R(x,y) = \text{det}(M(x,y)) - k \cdot (\text{trace}(M(x,y)))^2 \tag{3.14}$$

Here k is tunable parameter and mostly used as $k = 0.04$. This is an empirical value suggested in the literature. If $R(x, y)$ has a positive and local maxima response, then it is considered as a corner. Mostly a predefined threshold t is used where if $R(x, y) > t$, then that location is considered as a corner.

3.1.5. Shadow Detection

When the object is in front of a light source, shadow is created on the opposite side of the source. Even though shadows are not desired in some computer vision applications due to non-evenly distributed light on the image, for some applications shadows are evidence of object existence in the scene. Shadow also gives information on the geometry of the objects and their positions [111, 112].

In many cases, it is difficult to differentiate whether a pixel is a shadow point or only seen dark due to its texture and physical color characteristics. There are various methods in the literature to detect shadows from remotely sensed images. We use one of the most basic approach for shadow detection which is the histogram approach. When the histogram of a grayscale image is analyzed, the shadow regions have low illumination characteristics thus accumulated in the lower side of the grayscale histogram. So we use a basic Otsu's automated thresholding method to detect shadows in the image [113].

Detecting non-shadow regions as shadow is not crucial in our methods. The aim here is detecting as much shadow as possible. In the next sections, we will use the shadow pixels in feature vectors.

3.2. FEATURE VECTOR FORMATION

In this section, we will form feature vectors which will contain information of the vector's location, direction, and some other related data. Note that, these vectors are not meant to be extracted from 2D images only. They can also be extracted from height images.

In Figure 3.4 we illustrate some of the extracted feature sets. In this figure, we provide examples from left to right: on edge-based ribbon pairs, edge-based L-shapes, steerable filter based ribbon structures, steerable filter based L-shapes, and corners. Next, we will explain details on these.

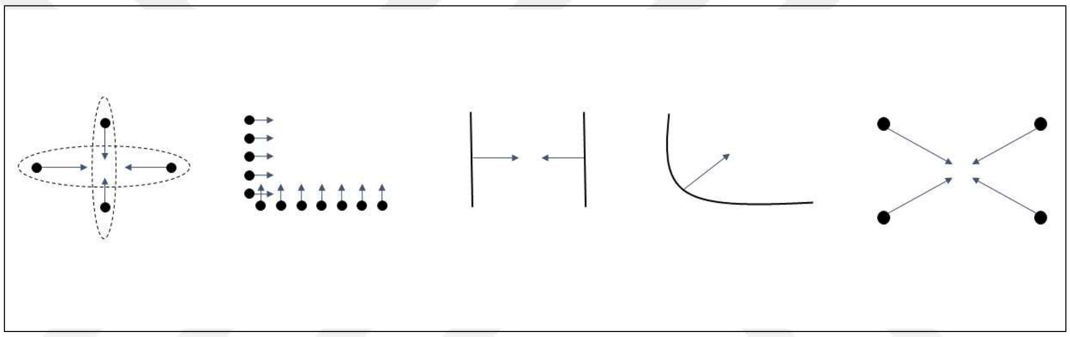


Figure 3.4. Extracted feature sets.

3.2.1. Edge-Based Ribbons

Suppose we extracted directional edge points for θ_n as described in Section 3.1.2 where (x_i, y_i) are the extracted edge locations which have gradient directions in interval of $[\theta_n - \alpha, \theta_n + \alpha]$ as described in Eqn. 3.4.

To extract a ribbon (point pairs), we detect parallel edge points that have opposite gradient directions. To do this, we extract edges (x_j, y_j) that are at opposite gradient direction with $[\theta_n + 180^o - \alpha, \theta_n + 180^o + \alpha]$. Then, for each point in (x_i, y_i), we find the nearest point in (x_j, y_j).

To do this, we define the search space as a cone like area as illustrated in Figure 3.5. In this figure, the black dots represent the extracted edge points. Also the dashed arrows represent

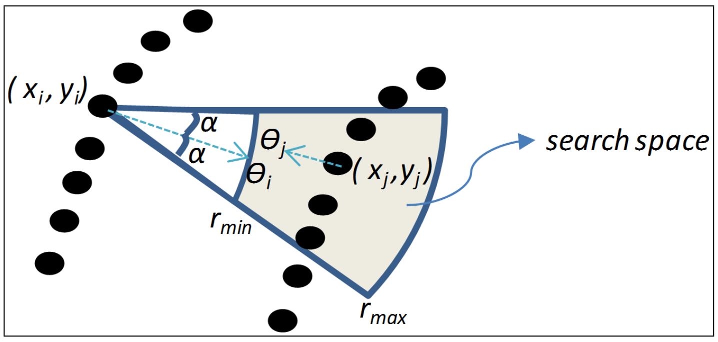


Figure 3.5. The search space for extracting directional edge based ribbon pairs.

the search direction. The search space in distance r and orientation ϑ is defined as

$$\begin{aligned} r_{\min} \leq & r \leq r_{\max} \\ \theta_n - \alpha \leq & \vartheta \leq \theta_n + \alpha \end{aligned} \tag{3.15}$$

where, $r_{\min}$ and $r_{\max}$ will be set as the smallest and largest object diameter measured in pixels in the image and α is the orientation tolerance.

Suppose the neighboring edge points satisfying the distance and orientation conditions (in Eqn. 3.15) to (x_i, y_i) are (x_j, y_j). Thus, the nearest point will be as $(x_n, y_n) = arg\min|(x_i, y_i) - (x_j, y_j)|$. Then, the edge point (x_i, y_i) and its nearest neighbor (x_n, y_n) can be used to set a center point $(\hat{x}_i, \hat{y}_i)$. This point will have the same distance to (x_i, y_i) and (x_n, y_n) with coordinates $\hat{x}_i = (x_i + x_n)/2$ and $\hat{y}_i = (y_i + y_n)/2$. This is the symmetry location of two points where we call it as the ribbon center. Now we will define a local feature vector that is describing the ribbon with its length σ_i and pairing strength, w_i, defined as

$$w_i = exp\left(-\frac{(180^o - \varphi_i)^2}{\rho}\right) \tag{3.16}$$

Here $\varphi_i = |\theta_i - \theta_j|$ is the angular direction difference of paired edge points. If the gradient direction of edge points are exactly opposite to each other ($\varphi_i = 180^o$), then $w_i = 1$. Otherwise, it will decay exponentially. In Figure 3.6 we give extracted ribbon centers with yellow marker. In this figure, red and blue markers are edge points (x_i, y_i) and (x_j, y_j), respectively.

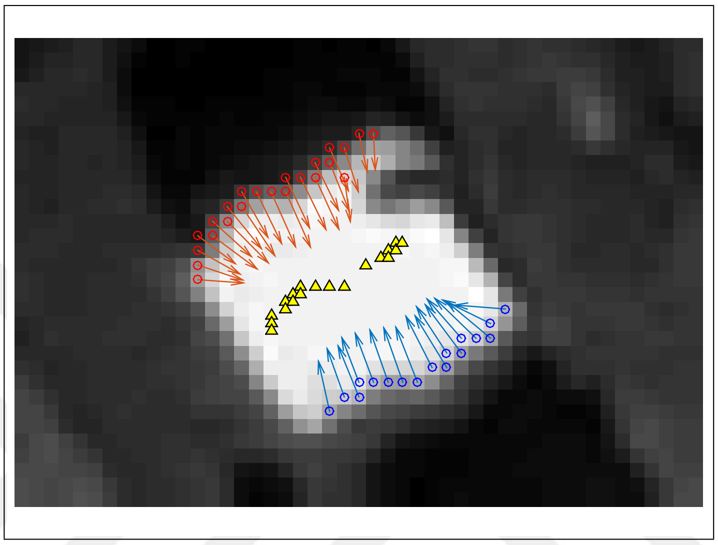


Figure 3.6. Extracted point ribbon centers for house image for $\theta_n = 60^o$ and $\alpha = 30^o$.

We define the length parameter, σ_i, as

$$\sigma_i = ||(x_i, y_i) - (x_n, y_n)|| \tag{3.17}$$

We extract all ribbon centers for N directions as $\theta\epsilon[0, \pi/N, 2\pi/N, ..., (N-1)\pi/N]$ and save all the descriptors in a feature vector as $\overrightarrow{r_1}(\hat{x}_i, \hat{y}_i, w_i, \sigma_i, \theta)$.

3.2.2. Edge Based L-Shapes

We extract L-shaped structures using directional edge points. An L-shaped structure can be used to model rectangular shapes. We illustrate this approach in Figure 3.7.

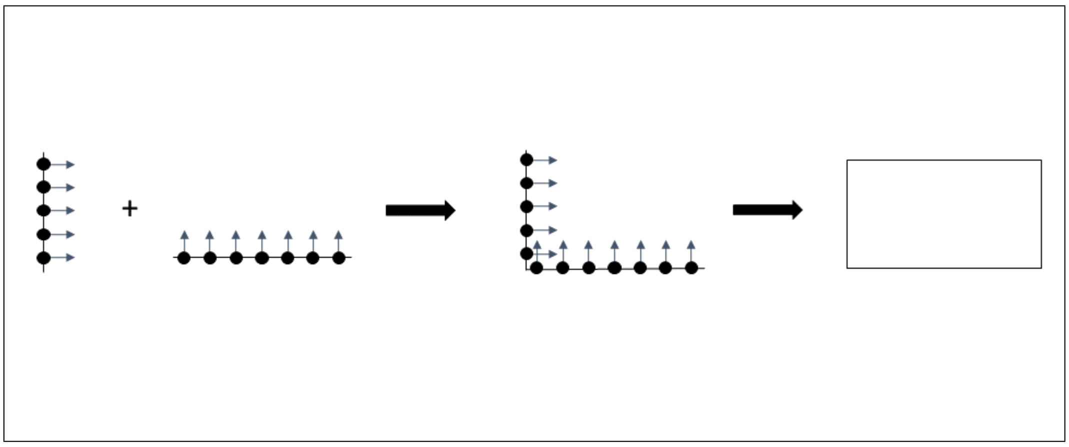
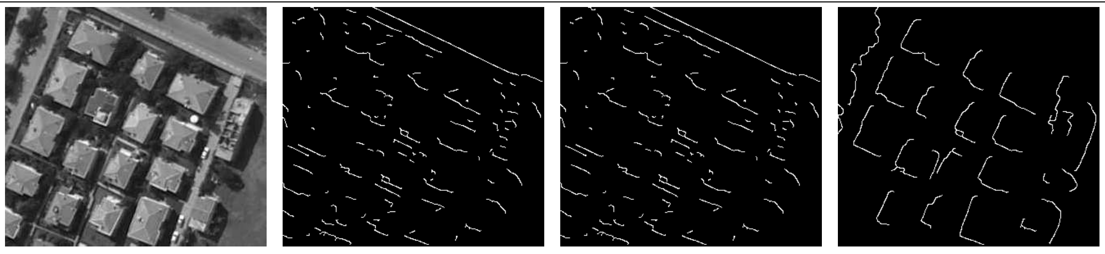


Figure 3.7. Two perpendicular ribbons for rectangle modeling.

In extracting edge based L-shapes, we first detect the edge of a grayscale image using Canny edge detector. We call the binary image with ones at edge locations and zeros at other locations as $B(x, y)$. Suppose we extracted directional edge points for θ_n and call this image as $B_1(x, y)$. Then, we select the edges with $\theta_n + 90^o$ directions and call this image as $B_2(x, y)$. Then, we use connected component analysis and combine these two binary images and select the edge segments that overlap with at least one pixel. Even though the edge points directions in two images apart with 90^o, we expect to see some overlapping at L-shape edges because of the direction interval defined with β as in Eqn. 3.4. We call the combined edges of binary image as $B_L(x, y)$. We give an example on an aerial image of buildings in Figure 3.8.



a. Aerial image of buildings. b. $B_1(x, y)$ c. $B_2(x, y)$ d. $B_L(x, y)$

Figure 3.8. L-shape feature extraction steps with directional edges.

As we detect L-shaped edge structures, we extract local feature vectors as follows. For each L-shaped edge segment, we find the location of end points and find the location of midpoint

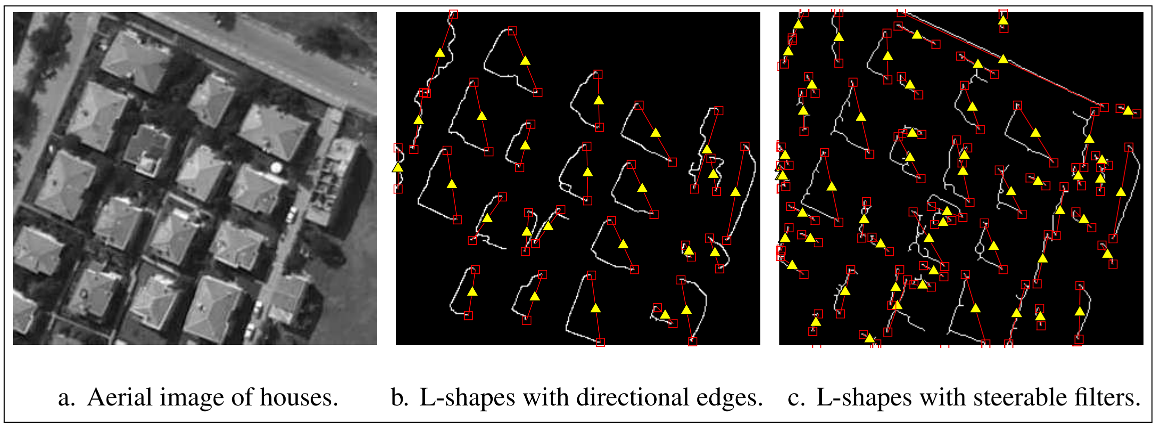

a. Aerial image of houses.
b. L-shapes with directional edges.
c. L-shapes with steerable filters.

Figure 3.9. L-shapes with directional edges and steerable filters.

of the edge segment by $\hat{x}_i = (x_i^{'} + x_i^{''})/2$ and $\hat{y}_i = (y_i^{'} + y_i^{''})/2$ where $(x_i^{'}, y_i^{'})$ and $(x_i^{''}, y_i^{''})$ are the location of the two endpoints of i^{th} L-shaped edge segment. In Figure 3.9.b, we give extracted endpoints and midpoints of L-shapes with red and yellow markers respectively.

We extract these L-shaped segments for N directions as $\theta \epsilon [0, \pi/N, 2\pi/N, ..., (N-1)\pi/N]$. Then we generate the feature vector $\overrightarrow{l_1}(i) = (\hat{x}_i, \hat{y}_i, w_i, \sigma_i, \theta)$ where we call it the edge based L-shape features vector. In this vector, w_i is the weight where we use it for discrimination of straight line segments from L-shaped segments. In Figure 3.8.d there are some straight line segments. Actually this happens due to the noisy edge pixels in the image.

Here, we define a weight for L-shape structures with the eccentricity value [114]. To do so, we calculate the eccentricity of the L-shaped curve. Eccentricity of a straight line segment is one. Eccentricity of a circle is zero.

We found that eccentricity of L-shaped curves are in the vicinity of 0.8. Therefore, we define a weight for L-shaped curves as

$$w_i = exp\left(-\frac{(ecc_i - 0.8)^2}{\rho}\right) \tag{3.18}$$

where ecc_i is the eccentricity of the i^{th} edge segment. We define the length parameter as

$$\sigma_i = ||(x_i^{'}, y_i^{'}) - (x_i^{''}, y_i^{''})|| \tag{3.19}$$

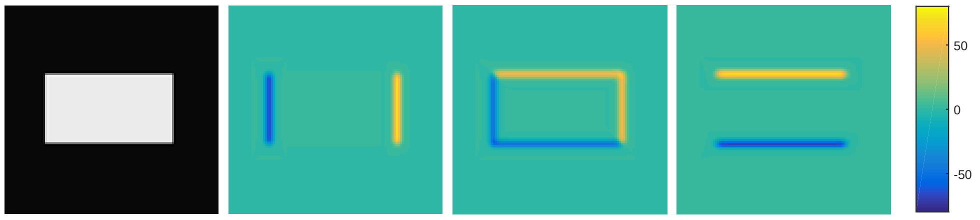
3.2.3. Steerable Filter Based Ribbons

In previous sections, we described ribbons using edge points. There, only two edge points could be paired as a ribbon. Here, ribbons are defined as line segments, not edge points. The line segments are extracted using steerable filters.

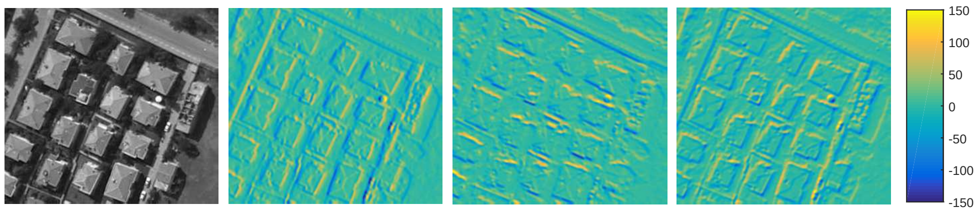
Steerable filters can be applied in any desired orientation. We apply steerable filter to a given image and obtain the filtered result J^θ as in Eqn. 3.9. According to the defined filtering direction θ, we expect to see high response perpendicular to filtering direction. We also see negative response from object to ground passing. In Figure 3.10, we give some examples. In this figure, we apply filtering in three directions as $\theta = 0^o$, $\theta = 45^o$, and $\theta = 90^o$ respectively. When $\theta = 0^o$ we see negative and positive responses from object to ground passing at the left and right side of the rectangle.

We threshold the filtering result and obtain some features. We define the thresholded results as $J_p^\theta = J^\theta > t$ and $J_n^\theta = J^\theta < -t$ where t is the threshold value and it might be obtained adaptively from J^θ. After thresholding, we apply connected component analysis and obtain center locations of connected components. Here we assume each connected component in J_p^θ and J_n^θ is a structural feature. The center location of the connected segments in J_p^θ and J_n^θ are (x_i, y_i) and (x_j, y_j) respectively.

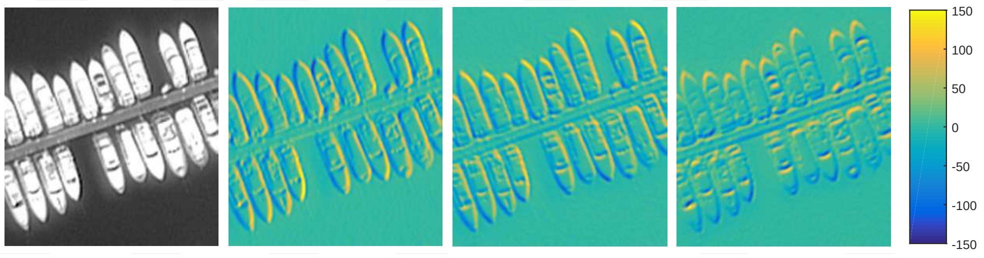
As in edge based ribbons, we extract possible ribbon centers by finding nearest points from (x_i, y_i) to (x_j, y_j) in a search space as described in Eqn. 3.15. Here we open a 2D cone like search space in the direction of filtering direction θ which was illustrated in Figure 3.5. As in edge-based ribbons, we find the nearest point and then find the ribbon center as



a. Rectangle



b. Building



c. Ship

Figure 3.10. Steerable filtering result for $\theta = 0^o$, $\theta = 45^o$, and $\theta = 90^o$ respectively.

$(\hat{x}_i, \hat{y}_i)$. We repeat this for multiple filtering directions and generate the feature vector $\overrightarrow{r_2}(i) = (\hat{x}_i, \hat{y}_i, w_i, \sigma_i, \theta)$ where we call it the steerable ribbon vector. Here σ_i is defined as

$$\sigma_i = ||(x_i, y_i) - (x_j, y_j)|| \tag{3.20}$$

and w_i is chosen to be constant for each ribbon.

3.2.4. Steerable Filter Based L-shapes

Steerable filters give flexibility of filtering direction of an image. We benefit from these to detect L-shaped structures of rectangular objects. If the filter direction is not exactly perpendicular to the edge alignment, then steerable filter function gives L-shaped curves for rectangular objects. We illustrated the approach in Figure 3.11. In this figure, θ denotes the steerable filter direction.

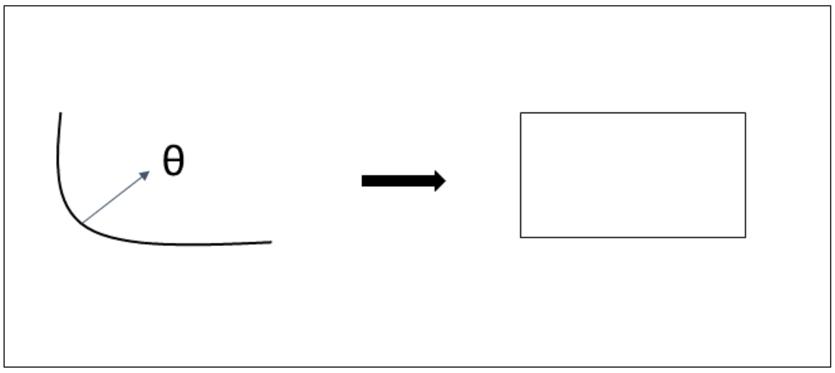


Figure 3.11. Steerable filter for L-shaped rectangle model.

To detect steerable filter based L-shapes, we first threshold the steerable filter result and obtain the binary image as $J_p^\theta = J^\theta > t$. We apply connected component analysis to J_p^θ and obtain the endpoints of each curve. Then, we find the midpoint of the curves by $\hat{x}_i = (x_i^{'} + x_i^{''})/2$ and $\hat{y}_i = (y_i^{'} + y_i^{''})/2$ where $(x_i^{'}, y_i^{'})$ and $(x_i^{''}, y_i^{''})$ are the location of the two endpoints of i^{th} L-shaped curve extracted from J_p^θ. We give the extracted midpoints on the same example

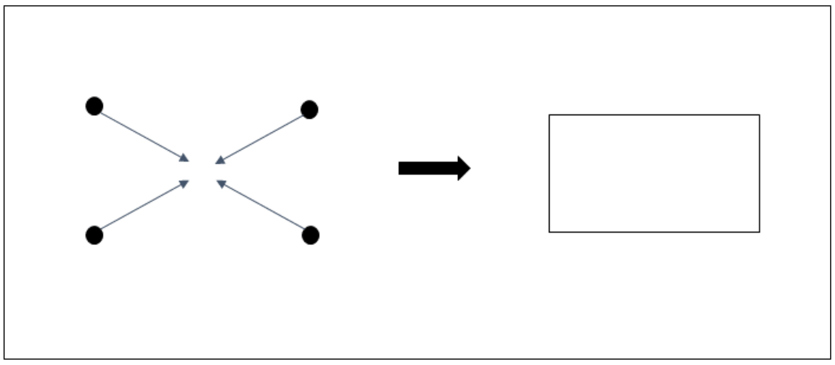


Figure 3.12. Corners for rectangle model.

image in Figure 3.9.c. We detected all of the L-shaped curves for $\theta = 45^o$. However, there are also individual straight lines. Analogous to the edge based L-shapes, we lower their effect with a weight as in Eqn. 3.18. Here, σ_i is the same with Eqn. 3.19. We apply filtering for N directions as $\theta\epsilon[0, \pi/N, 2\pi/N, ..., (N-1)\pi/N]$ and find L-shaped curves and generate the feature vector $\overrightarrow{l_2}(i) = (\hat{x}_i, \hat{y}_i, w_i, \sigma_i, \theta)$.

3.2.5. Corner Features

In this section, we generate a feature vector for each extracted corner in the image. We benefit from the gradient direction of corners where for bright objects the corner's directions are towards the object center. We illustrated this approach in Figure 3.12. We use Harris corner detector results and extract the location of the corner points as (x_i, y_i). Then, we shift every corner in the direction of θ_i as follows.

$$\begin{aligned} \hat{x}_i &= x_i + r_i sin(\theta_i) \\ \hat{y}_i &= y_i + r_i cos(\theta_i) \end{aligned} \tag{3.21}$$

where the corresponding gradient orientation is θ_i. r_i is the approximate shifting for locating the object center. Here r_i will be selected according to the desired object size. Then we

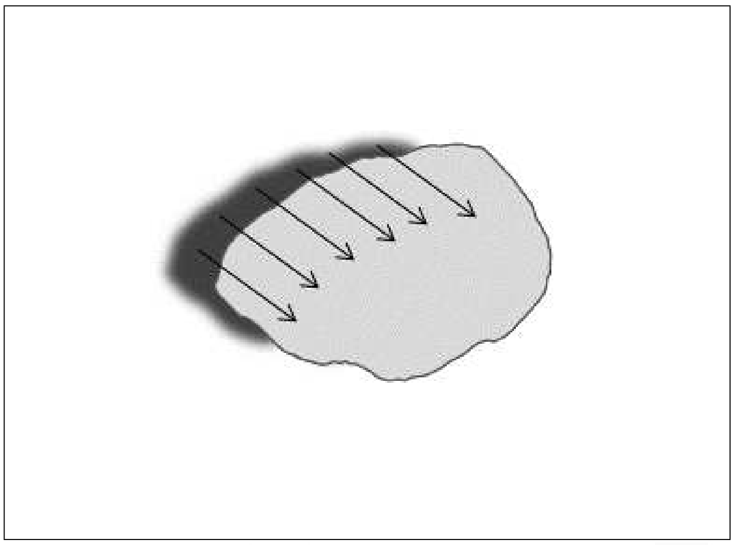


Figure 3.13. Shadow feature vectors illustration.

generate feature vector for each corner as $\overrightarrow{c}(i) = (\hat{x}_i, \hat{y}_i, w_i, \sigma_i)$. In this vector, w_i and σ_i will be constant, where the rationale selecting these will be explained in the experiments section.

3.2.6. Shadow Features

Shadow information (if available) can also be used to detect objects in a given image. It is evident that shadow may not be available due to the time of the day the image is taken or the weather conditions. Even though shadow itself doesn't give much information on the type of the object, it provides a strong clue on the location of an object if it can be extracted.

As explained in the previous section, we extract shadow segments from the grayscale image using Otsu's thresholding method. In doing so, we take a two level threshold such that the lowest level corresponds to shadow segments. The rationale here is as follows. If the image needs to be separated in two classes as shadow and not-shadow, then a single threshold is enough. If the background intensity is high compared to object intensity, then using a single threshold may cause wrong results in detecting shadows. Using a two level threshold for the grayscale image decreases the threshold compared to single level thresholding and gives better shadow detection results in satellite images.

Suppose we extracted shadow locations and saved them in a binary image $B_s(x, y)$ where $B_s(x_i, y_i) = 1$ for shadow locations and zero otherwise. Here (x_i, y_i) are the location of shadow points. To detect the object location, we shift the shadow points in the direction of sun's location. This information can be obtained from the metadata provided by the satellite image. In case of a missing metadata, the user can extract this information by observation. Let's assume that the direction of the sun is obtained as the yaw angle θ_t. Then, the shifted location becomes $(\hat{x}_i, \hat{y}_i)$ with

$$\begin{aligned} \hat{x}_i &= x_i + r_i \sin(\theta_t) \\ \hat{y}_i &= y_i + r_i \cos(\theta_t) \end{aligned} \tag{3.22}$$

We store the shifted locations in our feature vector $\overrightarrow{s}(i) = (\hat{x}_i, \hat{y}_i, w_i, \sigma_i, \theta_t)$. In this vector, w_i and σ_i will be constant, where the rationale selecting these will be explained in the experiments section.

3.2.7. Height Features

DSM data is obtained with LiDAR sensors and stereo image pairs and has been used for a few decades in remote sensing applications. In DSM, the height of surface is captured. Thus the height of objects on the ground is known. However, DSM data does not separate object and ground height measurements. It is basically the height of anything on the ground. In order to know the actual height of objects or terrain, one needs to filter objects or ground. This can be done by DTM generation which is achieved by finding correct ground points. For this matter, most algorithms in the literature try to filter out non-ground objects. The complexity of the environment and increasing spatial resolution of the data make it difficult to filter DSM data in an effective manner. Especially in urban areas, where there are dense and very large buildings, filtering non-ground objects thus DTM generation is challenging.

In this section, we are going to extract feature vectors on DSM data. These feature vectors later will be used for detecting objects on the ground. Due to changing terrain characteristics and object height, it is not easy to distinguish these two. When there is a change of elevation in DSM data, it is either an object (such as building, car, tree) or some kind of change on the ground (such as pavement, bridge) or the height of terrain itself is changing. In any case, the height change in DSM data may be taken as an evidence for existence of an object.

First we will find these changing points with an edge detector. Objects are assumed to be higher than their surrounding. Therefore, an edge point on the object boundary will have high and low elevation measurements at its surrounding. Assume we have detected the edges on DSM data and the location of edges are (x_i, y_i). Then, we find the maximum and minimum height of the (x_w, y_w) neighborhood around each of the corresponding edge point. Let's call DSM with I_s. Then, we define the neighborhood pixels on DSM image as

$$\begin{aligned} x_i - w/2 \leq x_w \leq x_i + w/2 \\ y_i - w/2 \leq y_w \leq y_i + w/2 \end{aligned} \tag{3.23}$$

where w is the window width. For each edge point, we find the location of maximum and minimum height in the neighborhood respectively with

$$\begin{aligned} (\hat{x_i}, \hat{y_i}) = argmax_{(x_w,y_w)} I_s(x_w, y_w) \\ (\check{x_i}, \check{y_i}) = argmin_{(x_w,y_w)} I_s(x_w, y_w) \end{aligned} \tag{3.24}$$

If the edge point is on an actual object, then the height difference is expected to be high. Otherwise it will be low. We define height difference as $\varDelta h_i = I_s(\hat{x_i}, \hat{y_i}) - I_s(\check{x_i}, \check{y_i})$. We store the maximum height locations and height differences in $\overrightarrow{h}(i) = (\hat{x_i}, \hat{y_i}, \varDelta h_i)$.

4. PROBABILISTIC OBJECT DETECTION

This chapter explains the probabilistic object detection method proposed in this study. Therefore it will start with explaining the general framework. Afterwards, details of the method will be explored.

4.1. GENERAL FRAMEWORK

Given an image or height data, we consider every pixel location as a candidate for being an object center. Thus for each pixel, we give an initial probability estimation of being a center of object. Then, using low and medium level features extracted from the image, we increase the probability at object center locations. This probability map is two dimensional where the size of it is the same with the image. In this map, every pixel location (x, y) has a probability of being the interested object center. In short, we model the object locations in an image as joint random variables and estimate their probability density function (pdf) using features.

Consider the toy image in Figure 4.1 where there are two houses with rectangular shape. Initially, we give equal probability for each pixel for being house center. Then, we extract some features in the image and utilize them to increase the probability at house centers. The desired probability map $P_B(x, y)$, probability map of building centers, is also given in Figure 4.1 on the right. In this example, we detect corner local features where each corner votes for the building center and increase the probability at vote locations.

In the above example, extracting corners is not sufficient for casting votes for building centers. We also need supplementary information of where and how to vote. Even though corners are strong evidences of existing of an object, it is not the only feature for locating objects. There are lots of other features may be used for voting. Based on the object descriptions and available data (panchromatic image, RGB image, DSM, Multispectral, etc.), we cast a condition on observations (features) and enforce the system to locate the objects.

We extract specific features from available data where each feature is indicating the presence

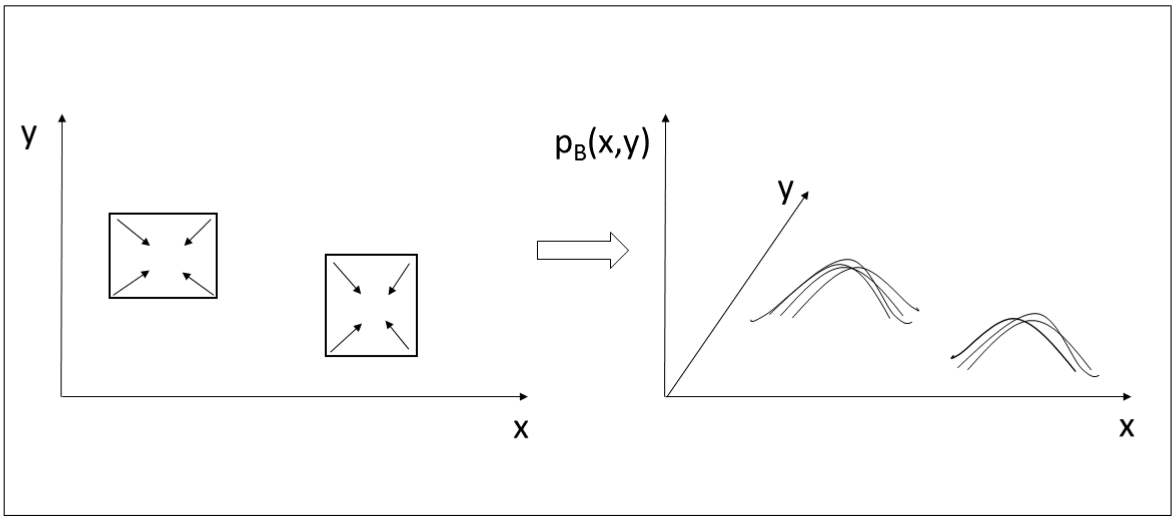


Figure 4.1. Desired pdf estimation of object center locations.

of a certain object. Some features exist if the specific objects exist and some features exist for more than one object type. We combine these features with probability maps where for each feature, we extract an individual probability map of object centers. Every feature is an evidence and used for updating the probability maps.

At first glance, one may believe this method is analogous to implicit shape modeling, Hough forests and generalized Hough transform [13]. However, there are very fundamental differences. Here, we differ from the implicit shape modeling and Hough forest methods with the training section. We do not use training sets for learning where to vote for each feature. In our approach, the training is implicit within the shape model. We assume that objects have either simple shape or combination of simple shapes. We define specific rules for each object and enforce the features with these rules while voting for object centers. Actually, when the certain shape of some objects are considered in satellite images such as house, it is difficult to train the features for all cases of house images. Other training based methods try to separate parts of a specific object and train the parts to find the object. However, as in the house image case where it is simply a rectangular shape, it is difficult to define individual parts for training. Indeed, we use general observations on simple shapes and use them for probabilistic voting. Generalized Hough transform finds the parameters of the system, where

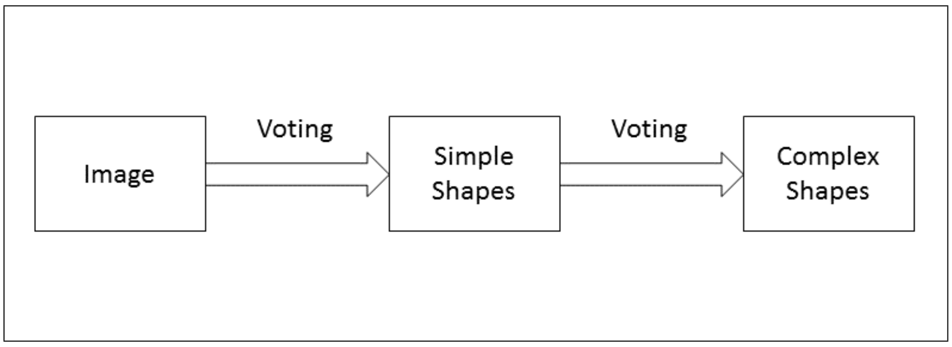


Figure 4.2. Proposed high level framework for object detection from satellite images.

we describe the object and search for the object centers directly. Thus, we do not try to find any parameters for the object.

We assume there are two types of object classes in satellite images: simple and complex shapes. We place buildings, trees and ships in the simple shapes and cars and airplanes in the complex shapes category. These objects are our focus of interest in the satellite images and other data. A very high level of object detection framework was given in Figure 4.2.

For instance, we assume buildings as rectangular shapes and cars as combination of two or three rectangular shapes. So if we would like to locate the cars in the image, we first find rectangles and then find appropriate combination of rectangles. Of course, the shape is not the only feature for discriminating the objects. We also benefit from the shadow, height and size constraints of objects. To give an example, consider again the cars where the spatial size and height are well defined. So, in the probabilistic voting process our rules of voting for object center depends on

- Observations (edge, corner, line, gradient, shadow, etc.)
- Shape constraint
- Spatial size constraint
- Height constraint
- Geometric structure constraint

In our object detection and segmentation framework, we always try to vote for most possible object center. As mentioned before, for each feature vector we extract individual probability maps. Assume that we have detected some number of features $f_1, f_2, ..., f_n$. Here, the sub index number defines the different feature vectors such as corner, line, or shadow. For each feature, we extract probability maps $p_O(x, y|f_1), p_O(x, y|f_1), ..., p_O(x, y|f_n)$. Here sub index O defines the object type. We combine these probability maps in a decision fusion step and finalize the probability map for the objects $P_O(x, y)$. After detecting the object centers with the final probability map, we extract shapes of objects which is the subject of next chapter.

4.2. DETECTION OF SIMPLE SHAPED OBJECTS

Many object features are discovered with bare observations. Although these observations are simple, they become powerful tools for distinguishing certain objects. Since houses in satellite images are mostly in rectangular shapes, it is a fundamental process to distinguish rectangular shapes. There are other objects in the scope of our interest such as trees, ships, cars, and airplanes. Cars also have rectangular outline. On the other hand, ships and trees have ellipsoidal shapes. They resemble crude rectangular shapes. The other way around is also true where in a bad resolution image, houses look like in circular shapes.

In our object detection framework, we try to find the object centers in a given image. For this reason, in our models we benefit from the extracted features and try to locate the object center with probabilistic voting. We already gave information on the feature vectors in the previous chapter. In this section, we will define models of the objects according to their geometrical shapes.

We are interested in detecting simple and complex shapes in satellite images. Buildings, trees and ships have simple shapes whereas cars and airplanes have complex shapes in our models. We give such examples in Figure 4.3.a and Figure 4.3.b. We represent buildings with rectangles and trees and ships with ellipsoidal shapes. Cars and airplanes have complex but well defined shapes where we model them as combination of multiple rectangles. We give some example images in Figure 4.3.c.

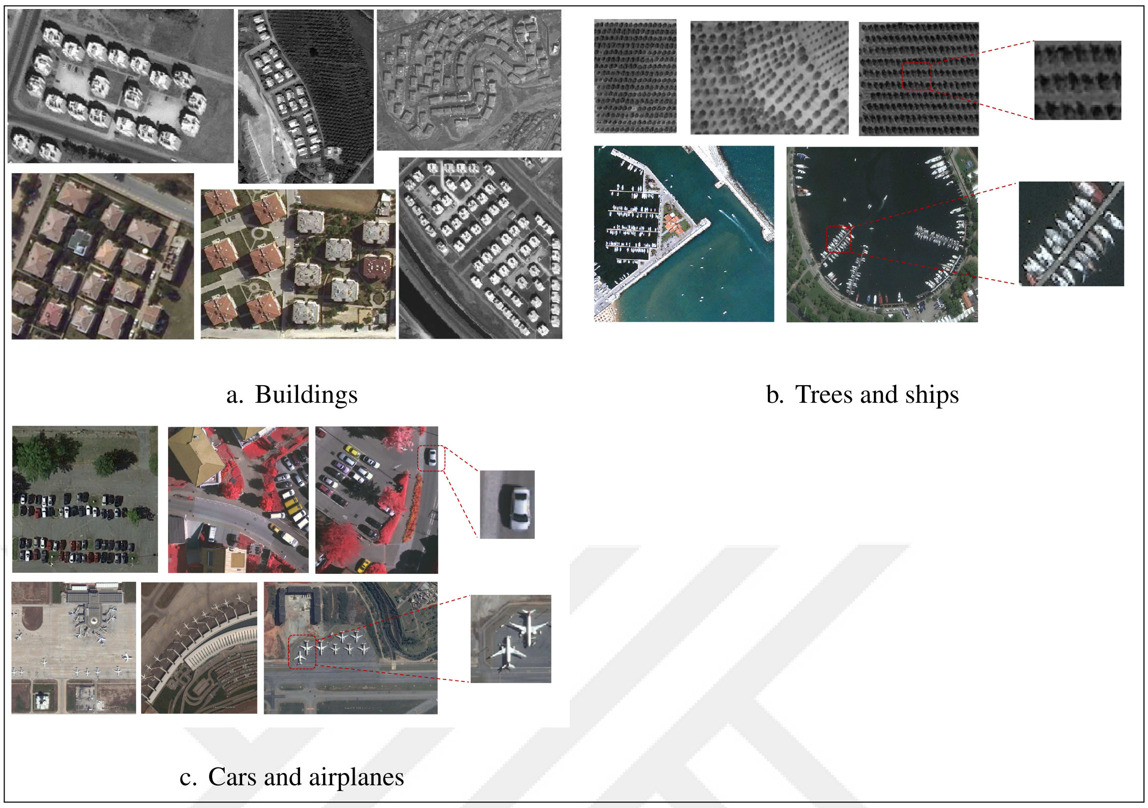


Figure 4.3. Simple and complex shape object samples from satellite images.

4.2.1. Detection using Single Feature Set

In this section, we will talk about the voting for the object centers where we assume our object have simple shapes. We will benefit from the extracted features in the previous chapter. We will define some rules and select features to locate object centers.

In the previous chapter, we extracted some feature vectors. These feature vectors are $F = \left\{\overrightarrow{r_1}, \overrightarrow{l_1}, \overrightarrow{r_2}, \overrightarrow{l_2}, \overrightarrow{c}, \overrightarrow{s}, \overrightarrow{h}\right\}$. For instance, if someone wants to detect rectangular objects with only a single feature set, such as edge based L-shapes only ($F = \left\{\overrightarrow{l_1}\right\}$), then we can model the center locations of rectangular objects in the image as joint random variables and estimate their probability density function (pdf).

For each feature vector, we extracted locations, orientation, weight and length parameters. For example, for L-shapes from edge points, we defined the feature vector as $\overrightarrow{l_1}(i) =$

$(\hat{x_i}, \hat{y_i}, w_i, \sigma_i, \theta)$. Here we extracted the vectors for multiple orientations of θ. While extracting the feature vectors, we tried to find most possible convex object centers. We assume that every feature votes for the possible object center and thus at object centers the votes accumulate. So we use these parameters in a joint kernel density estimation. First, we define the pdf for single orientation as below

$$p_{O,\theta}(x, y|F) = \sum_{i=1}^{K} w_i \exp\left(-\frac{(x - \hat{x_i})^2 + (y - \hat{y_i})^2}{2\sigma_i}\right) \tag{4.1}$$

where in this case $O = rectangle$, $F = \left\{\overrightarrow{l_1}\right\}$ and K_i is the total number of feature vectors. Using this, we write the following sum to obtain pdf for all orientations

$$p_O(x, y|F) = \sum_{\theta} p_{O,\theta}(x, y|F) \tag{4.2}$$

It is hypothesized that the mode locations of $p_O(x, y|F)$ are possible rectangle center locations. Alternatively, one can use other feature vectors and estimate pdf for each feature set separately. Note that corner feature vectors, $\overrightarrow{c}$, are extracted for all orientations at once. Thus, the summation in Eqn. 4.2 is not necessary. It is also true for shadow feature vectors $\overrightarrow{s}$ and height feature vectors $\overrightarrow{h}$.

4.2.2. Detection using Multiple Feature Sets

In images, we encounter with many different cases of objects. Sometimes only one or two edges of rectangular objects are visible due to illumination, noise, color indifference of object and ground or spatial resolution of the image. Such examples are given in Figure 4.4 where some object boundaries are not clear. So we are examining different types of attributes of

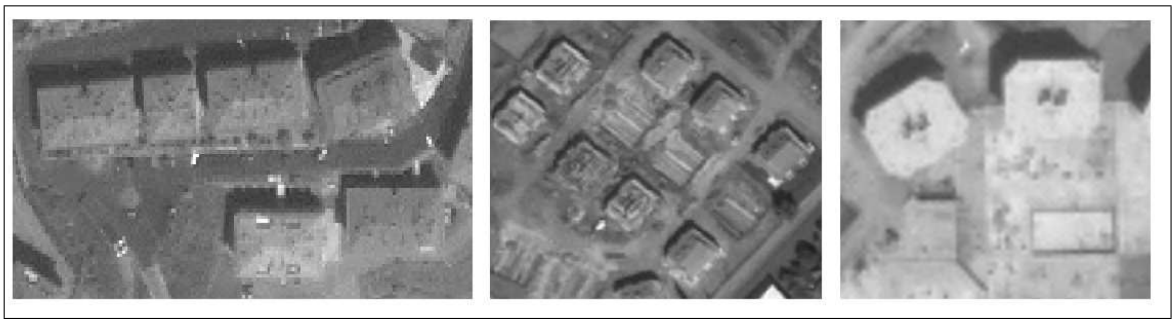


Figure 4.4. Noisy and low resolution panchromatic images of buildings.

rectangular shapes to handle such problems.

We can use shadow and height features for any type of objects in the image. Other than these features, objects might be modeled according to their simplest shapes. As mentioned in the previous section, rectangular shapes are modeled with combination of ribbons, edge based L-shapes, steerable filter based L-shapes, and corners. The hierarchical structure for rectangular shapes is given in Figure 4.5.

Similarly, we can model ellipsoidal shapes with combinations of ribbons and corners. The hierarchical structure for ellipsoidal shapes is given in Figure 4.6. The extracted single feature set is not meant to be used to detect rectangles or ellipses only. The motivation behind using these feature sets is using the combinations of multiple feature sets which lead to detect specific simple shapes in images. In this section, we will give the details of combination of feature sets to estimate a finer pdf for building, tree and ship detection.

Let's call the extracted feature vectors as $F = f_1, f_2, ..., f_N$. Given these feature vectors, we want to locate the object centers using the pdf estimation as

$$p_O(x, y|F) = p(\Theta|f_1, f_2, ..., f_N) \tag{4.3}$$

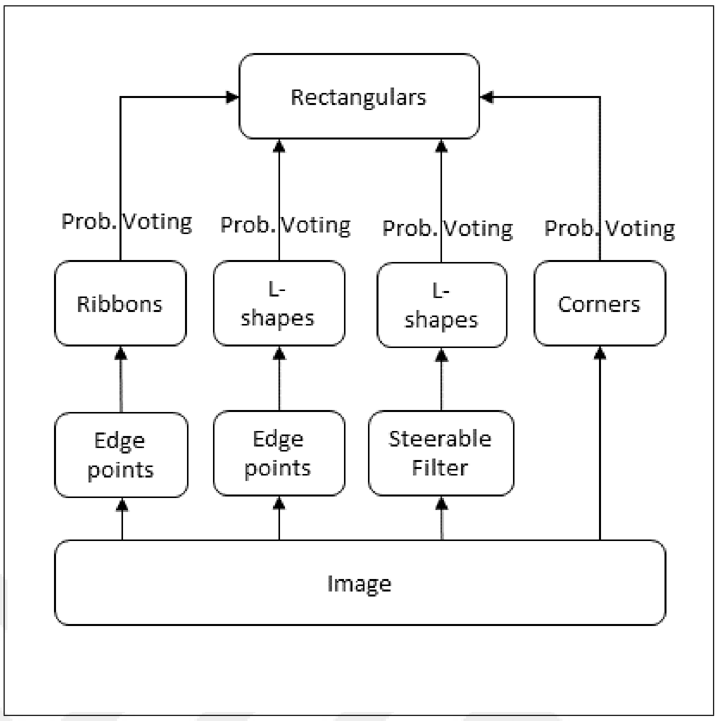


Figure 4.5. Detecting rectangular shapes using multiple features.

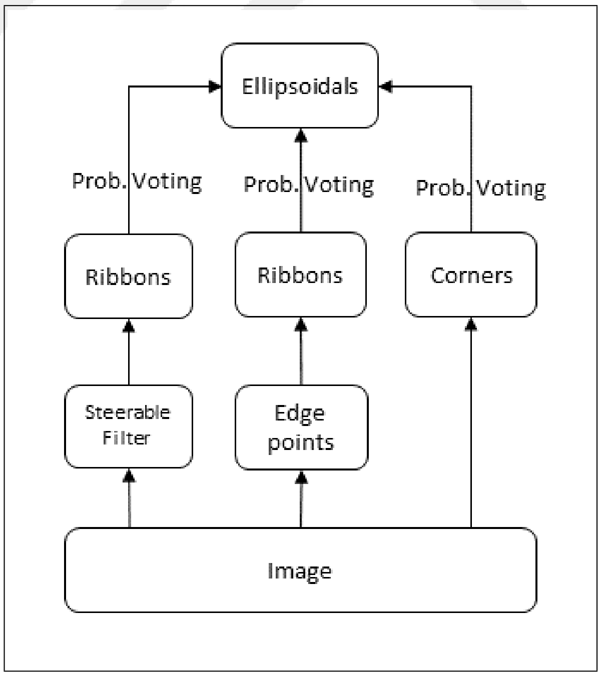


Figure 4.6. Detecting ellipsoidal shapes using multiple features.

where Θ is the object center locations. Bayes theorem says that initial probability of an hypothesis might be updated based on the related evidences [115]. Bayes assumption for hypothesis H and evidence E is

$$p(H|E) = \frac{p(E|H)p(H)}{p(E)} \tag{4.4}$$

In our case, the prior distribution of object centers is $p(\Theta)$ and the posterior pdf is $p_O(\Theta|f_1, f_2, ..., f_N)$. Using the Bayes theorem, we can write the posterior pdf estimation as

$$p(\Theta|f_1, f_2, ..., f_N) = \frac{p(f_1, f_2, ..., f_N|\Theta)p(\Theta)}{p(f_1, f_2, ..., f_N)} \tag{4.5}$$

We want to determine the Θ values maximizing this equation. In this equation, the denominator $p(f_1, f_2, ..., f_N)$ is independent of Θ. Assuming the conditional independence

$$p(f_i, f_j|\Theta) = p(f_i|\Theta)p(f_j|\Theta) \tag{4.6}$$

we can rewrite Eqn. 4.5 as

$$p(\Theta|f_1, f_2, ..., f_N) = \prod_{i=1}^{N} p(f_i|\Theta)\frac{p(\Theta)}{p(f_1, f_2, ..., f_N)} \tag{4.7}$$

Using the Bayes theorem again

$$\begin{aligned} p(\Theta|f_1, f_2, ..., f_N) &= \prod_{i=1}^{N} \frac{p(\Theta|f_i)p(f_i)}{p(\Theta)} \frac{p(\Theta)}{p(f_1, f_2, ..., f_N)} \\ &= \frac{1}{p(\Theta)^{N-1}} \frac{p(f_1)p(f_2)\cdots p(f_N)}{p(f_1, f_2, ..., f_N)} \prod_{i=1}^{N} p(\Theta|f_i) \end{aligned} \tag{4.8}$$

Here, when we assume the prior distribution of object center locations $p(\Theta)$ is uniformly distributed within the image. Moreover, noticing the terms before the product term are independent of Θ, then the maximum likelihood (ML) estimator (thus the Bayesian estimator) is obtained by maximizing the function below wrt Θ.

$$p(\Theta|f_1, f_2, ..., f_N) \propto \prod_{i=1}^{N} p(\Theta|f_i) \tag{4.9}$$

Thus, instead of fully estimating $p(\Theta|f_1, f_2, ..., f_N)$, we estimate $\prod_{i=1}^{N} p(\Theta|f_i)$ and locate the possible object centers $\widehat{\Theta}$ by

$$\widehat{\Theta} = \arg\max_{\Theta} \prod_{i=1}^{N} p(\Theta|f_i) \tag{4.10}$$

In the previous chapter, we extracted the following feature vectors.

- f_1: $\overrightarrow{r_1}$, edge based ribbon feature vectors.
- f_2: $\overrightarrow{l_1}$, edge based L-shape feature vectors.
- f_3: $\overrightarrow{r_2}$, steerable filter based ribbon feature vectors.

- f_4: $\overrightarrow{l_2}$, steerable filter based L-shape feature vectors.
- f_5: $\overrightarrow{c}$, corner based feature vectors.
- f_6: $\overrightarrow{s}$, shadow based feature vectors.
- f_7: $\overrightarrow{h}$, height based feature vectors.

To detect typical objects in remote sensing images, we will use combinations of these features. If shadow is not present, then it won't be used. Similarly, if the height data is not available, it won't be used. Next, we explain which features will be used for which objects.

4.2.2.1. Building Detection

We extract four features for rectangular shapes and for each of them we estimate pdfs of possible object centers. We also estimate pdf for shadow and height of any type of object. While extracting those features for buildings, we use some size constraints. For example, while extracting edge based ribbons, the search range of ribbon points are set according to minimum and maximum building size. Similar constraints are also utilized for other feature vectors. These will be explained in more detail in the experiments section.

If there isn't shadow in the image, or there isn't any height data available, then we will only use the subset of the feature vectors. For instance, when we use only (f_2, f_3, f_4) features for building detection, then the function to be estimated for building centers will be

$$p_B(x, y|f_2, f_3, f_4) \propto p_B(x, y|f_2)p_B(x, y|f_3)p_B(x, y|f_4) \tag{4.11}$$

Then, we will find the modes of the pdf and assume that these are possible building centers. In the experiments section, we will show that utilizing every feature greatly reduces false alarms, but also increases missed detections.

4.2.2.2. Tree and Ship Detection

Trees and ships have ellipsoidal shapes. So, we use the following feature vectors.

- f_1: $\overrightarrow{r_1}$, edge based ribbon feature vectors.
- f_3: $\overrightarrow{r_2}$, steerable filter based ribbon feature vectors.
- f_5: $\overrightarrow{c}$, corner based feature vectors.
- f_6: $\overrightarrow{s}$, shadow based feature vectors.
- f_7: $\overrightarrow{h}$, height based feature vectors.

While extracting the features for ships and trees, we use size constraints. For example, while extracting edge based ribbons for trees, the search range of ribbon points are set according to minimum and maximum tree size. Similar constraints are also utilized for ships and other feature vectors. These will be explained in more detail in the experiments section.

However, using shadows to detect ships mostly doesn't work due to the closely located ships in harbor. Also finding height data for ships is difficult. Similarly, because of the illumination effects in the image and spatial resolution, trees are not detectable using steerable filters. So, one needs to select a subset of these features to detect ships and trees.

4.3. DETECTION OF COMPLEX OBJECTS

In the previous section, we detected simple shaped objects with combination of different features in a probabilistic framework. Here, we will detect complex objects such as airplanes and cars with combination of simple shapes.

4.3.1. Airplane Detection

We model an airplane with three main parts: body, right wing, and left wing. In our model, the main parts are simple rectangles. It is also true to assume that parts have ribbon like structures since the part lengths are very large compared to their width. We define a geometrical relationship between the parts to detect airplane objects in the image. Each part will vote for the center location of airplane.

The model is given in Figure 4.7. An airplane has a perfectly symmetric shape and the symmetry point is the center of its body. In our model we benefit from its symmetrical shape.

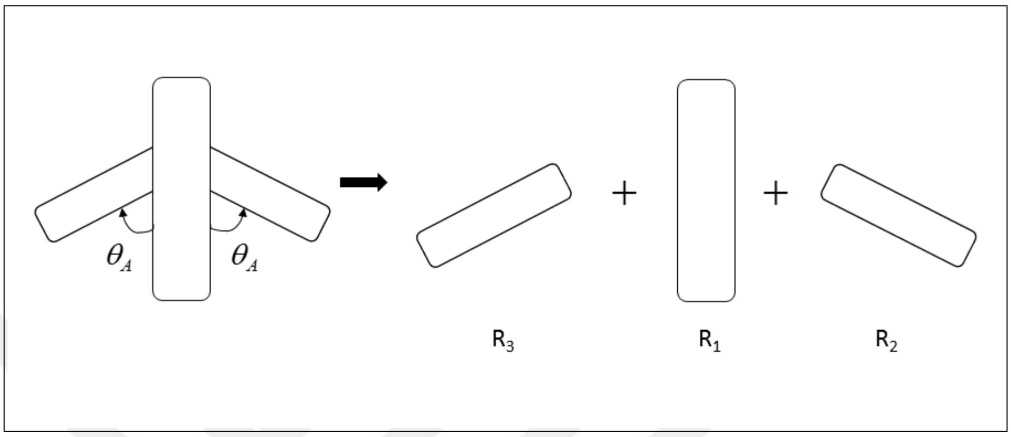


Figure 4.7. Multiple rectangles model for airplanes.

We give a sample airplane satellite image in Figure 4.8.a. We also provide the extracted Canny edges with their gradient direction in Figure 4.8.b. Let's make some observations on this image. The first observation is the angle between the body and the wings where it is same for both wings and we call it θ_A.

The second observation is the edge direction of the airplane relative to its center. Let's have a look only to the right wing. It has a ribbon like structure. The relative angle from the center of the right wing to the airplane center is perpendicular to directions of wing edges. It is also true for other wing and the main body part of airplane. So in our model we use these two observations.

In Figure 4.9, we give the geometric relationship between the parts of the airplane and voting locations of the parts. We do not know the orientation of the airplane. Thus, we benefit from the relationship of the body and its wings.

As illustrated in Figure 4.9, the orientation of body, θ_1, and two wings, θ_2 and θ_3, can be given as follows

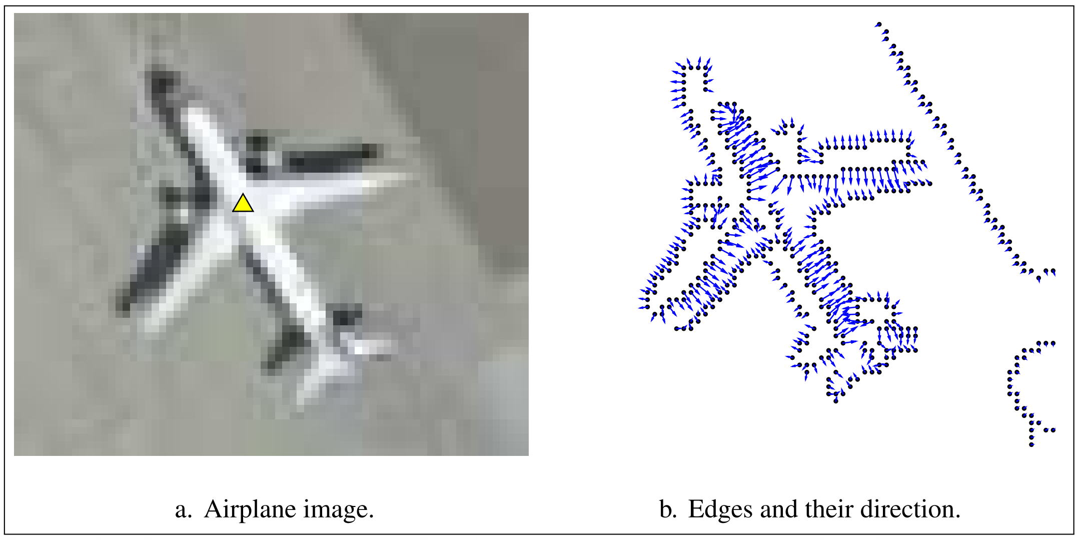


Figure 4.8. Sample satellite image of airplane and Canny edges.

$$\theta_1 = \theta_i$$

$$\theta_2 = \theta_i + \theta_A \tag{4.12}$$

$$\theta_3 = \theta_i + 180^o - \theta_A$$

In Figure 4.9 the edge points and possible center of parts are also plotted. We only know that the relative angle from center of the airplane to the part centers (ribbon center) is perpendicular to edge directions. Thus, for each edge point pair (ribbon), we vote for two possible airplane locations. For example for the body part of the airplane, even though we find a possible part center, we do not know if the center of airplane is below or above the part center. So, we vote for both directions as illustrated in Figure 4.9.

Assume for a specific direction $\theta_1 = \theta_i$, we extracted edge pairs. This is shown as black dots in Figure 4.10.a. Then, using edge pairs we extract body part centers which is shown as black squares in Figure 4.10.b. Their center location are denoted as $(\hat{x}_i, \hat{y}_i)$. Since we paired

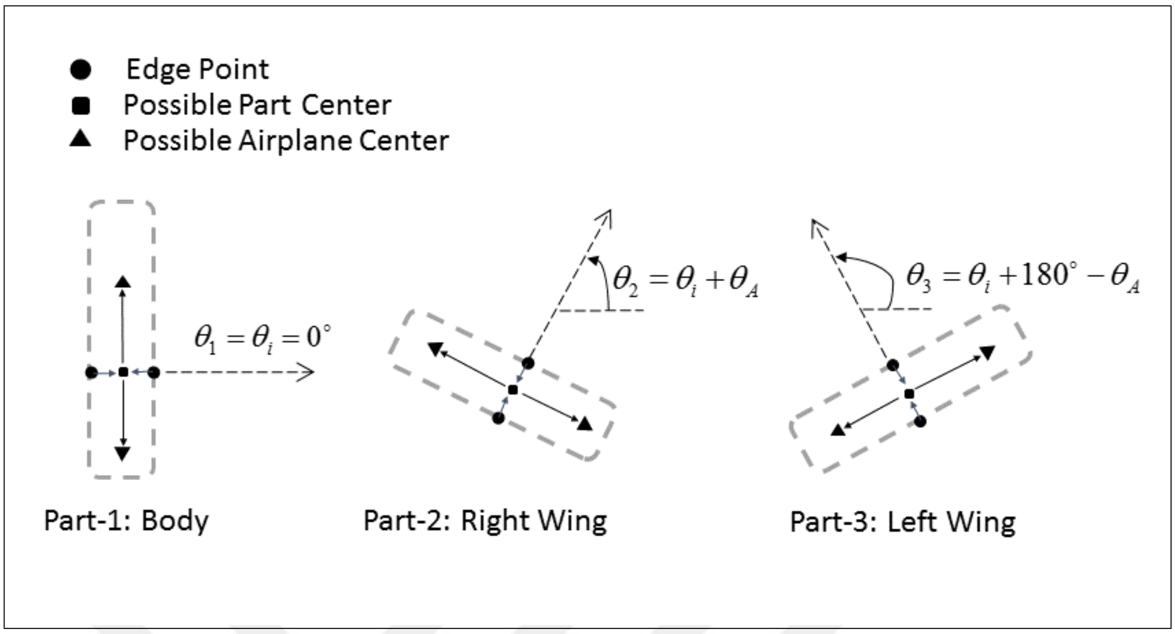

● Edge Point
■ Possible Part Center
▲ Possible Airplane Center
$\theta_1 = \theta_i = 0^\circ$
$\theta_2 = \theta_i + \theta_A$
$\theta_3 = \theta_i + 180^\circ - \theta_A$
Part-1: Body
Part-2: Right Wing
Part-3: Left Wing

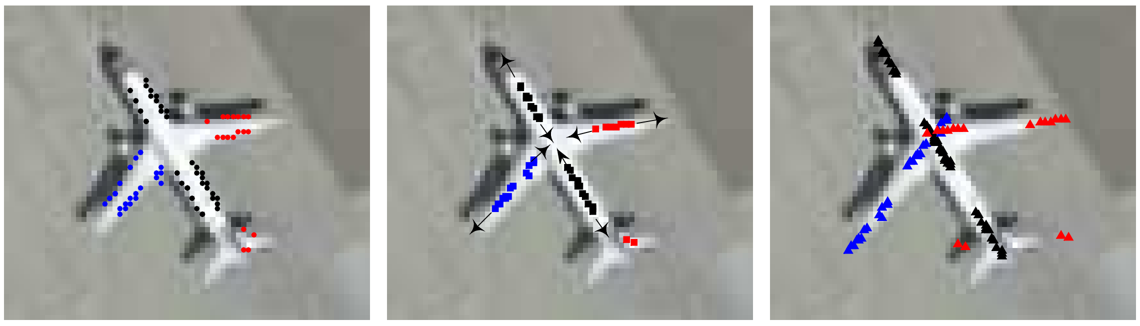
Figure 4.9. metry of airplane parts and possible voting locations.

each edge point, there are multiple center points for the body part of airplane. As shown in Figure 4.10.b, the extracted centers will vote to perpendicular directions of the edge directions. This can be described as follows.

$$\begin{aligned} \bar{x}_i &= \hat{x}_i \mp r\cos(\theta_1 + 90^o) \\ \bar{y}_i &= \hat{y}_i \mp r\sin(\theta_1 + 90^o) \end{aligned} \tag{4.13}$$

which is equivalent to

$$\begin{aligned} \bar{x}_i &= \hat{x}_i \pm r\sin(\theta_1) \\ \bar{y}_i &= \hat{y}_i \mp r\cos(\theta_1) \end{aligned} \tag{4.14}$$



a. Extracted edges for θ_1 in black, θ_2 in red and θ_3 in blue markers.

b. Possible part centers and voting directions.

c. Possible airplane centers.

Figure 4.10. Voting method for airplane parts.

In Eqn. 4.14, r is the voting distance and $(\bar{x}_i, \bar{y}_i)$ are possible airplane centers. We do not know the distance of the part center to airplane center. We choose a constant value for this parameter. There are three reasons for that. First, there are multiple body centers (ribbon centers) located side by side. So, we have a good chance that at least one of the body center point will vote for exactly the airplane center. Second, the size of the airplanes do not change as much as other objects such as buildings. So we can define the voting distance using size constraints. Third, we use a probabilistic voting scheme, thus the uncertainty of the voting distance is expected to be handled by the method.

We repeat these steps for right wing, θ_2, and left wing, θ_3. The results for other parts are shown in Figure 4.10.a and Figure 4.10.b. In Figure 4.10.c we show the voting locations. As can be seen from this image, the votes of body and wings of airplane are gathered at the airplane center. We use a joint probabilistic model to locate the center of airplane as in simple shaped object detection case as follows.

$$p_{A,\theta_i}(x, y|R_1, R_2, R_3) \propto p_{A,\theta_i}(x, y|R_1) p_{A,\theta_i}(x, y|R_2) p_{A,\theta_i}(x, y|R_3) \qquad (4.15)$$

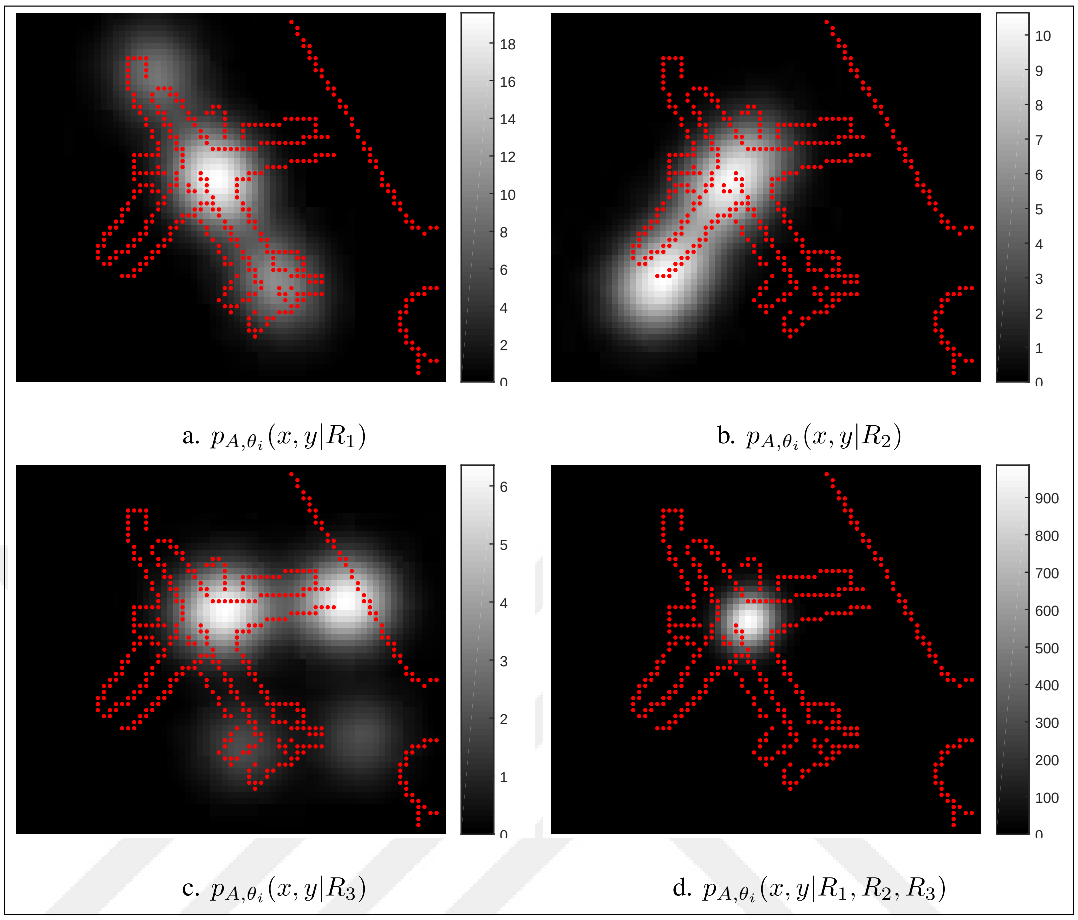


Figure 4.11. Airplane center pdf estimates for $\theta_i = 30^o$.

where R_1, R_2 and R_3 are the extracted part centers (ribbon centers). We define $p_{A,\theta_i}(x, y|R_1)$ as

$$p_{A,\theta_i}(x, y|R_1) = \sum_{i=1}^{N_i} \exp\left(-\frac{(x - \bar{x}_i)^2 + (y - \bar{y}_i)^2}{2\sigma_i}\right) \tag{4.16}$$

For the other two parts, these equations become

$$p_{A,\theta_i}(x,y|R_2) = \sum_{j=1}^{N_j} \exp\left(-\frac{(x-\bar{x_j})^2+(y-\bar{y_j})^2}{2\sigma_j}\right) \tag{4.17}$$

$$p_{A,\theta_i}(x,y|R_3) = \sum_{k=1}^{N_k} \exp\left(-\frac{(x-\bar{x_k})^2+(y-\bar{y_k})^2}{2\sigma_k}\right) \tag{4.18}$$

where the part centers for θ_2 are $(\hat{x_j}, \hat{y_j})$ and the part centers for θ_3 are $(\hat{x_k}, \hat{y_k})$. The part centers again are given in Figure 4.10.b with black, red an blue square markers.

The part center locations are calculated as below

$$\begin{aligned} \bar{x_j} &= \hat{x_j} \pm r\sin(\theta_2) \\ \bar{y_j} &= \hat{y_j} \mp r\cos(\theta_2) \end{aligned} \tag{4.19}$$

$$\begin{aligned} \bar{x_k} &= \hat{x_k} \pm r\sin(\theta_3) \\ \bar{y_k} &= \hat{y_k} \mp r\cos(\theta_3) \end{aligned} \tag{4.20}$$

In Figure 4.11 we give the vote maps. In Figure 4.11.d the final estimated pdf for airplane center location has high probability at its correct airplane center.

Even though we have a good result, we assumed airplane's body orientation is $\theta_1 = \theta_i$. We

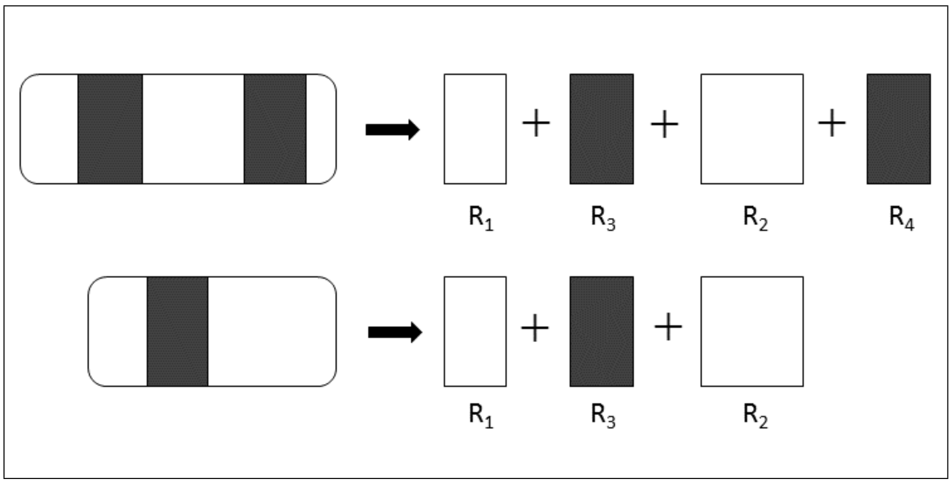


Figure 4.12. Multiple rectangles model for cars.

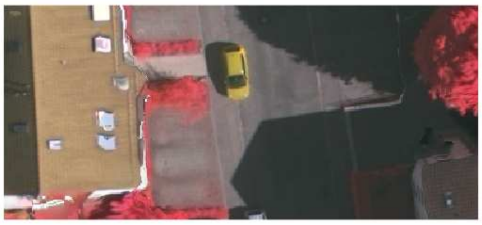
need to find the airplanes that are in other orientations also. So we marginalize the pdf in Eqn. 4.15 for $\theta\epsilon[0, \pi/N, 2\pi/N, ..., (N-1)\pi/N]$ and estimate the final pdf for all airplanes as follows.

$$p_A(x, y|R_1, R_2, R_3) \propto \sum_{i=1}^{N} p_{A,\theta_i}(x, y|R_1, R_2, R_3) \tag{4.21}$$

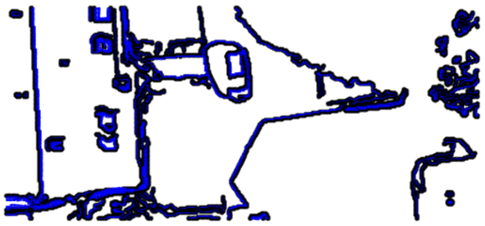
Then, we will find the modes of the final estimated pdf and assume the modes are possible airplane centers.

4.3.2. Car Detection

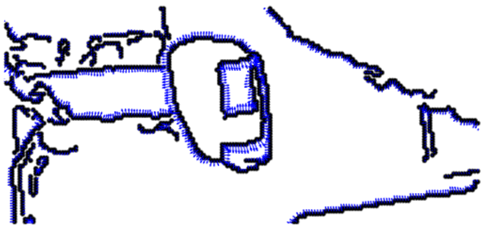
In 2D images, cars are seen as rectangles side by side. If we have 3D data, we can use height and size constraints based probabilistic voting method. Cars have almost standard dimensions. For example, a midsize car has typical dimensions of 1.7 m $\times 1.5$ m $\times 4.2$ m in width, height, and length respectively. Having high resolution images and height data makes it possible to detect such objects using size constraints. The rectangle model for cars is given in Figure 4.12.



a. Car image.



b. Edges and their direction.



c. Edges and their direction, zoomed

Figure 4.13. Sample satellite image of car and Canny edges.

The majority of car shapes will be subset of one these two models in Figure 4.12. In both sedan and station model cars, the common model would be the one with three rectangles. In this model, the windshield is rectangular and almost always in a dark color. On the front end and back end of the car there are rectangles again. In our approach, we will find the rectangles in the image in which only the relative positions of these rectangles fit the car model will be remained.

In Figure 4.13.a a sample satellite image of a car is given where the car is on the road next to a building. There are shadows and trees as well in the image. We draw the Canny edges and their gradient orientation in Figure 4.13.b. The zoomed version is given in Figure 4.13.c. There are lots of clutter in the image. Even though one side of the windshield could not be detected in the edges, the three rectangles model for the car is easily noticed.

In Figure 4.14, we illustrate how the voting process is implemented. In the voting process, the rectangles always vote for the possible windshield location. Assuming the car is in the direction of south (90^o), first we find the back end rectangle ($\theta_1 = 0^o$). The car's length might

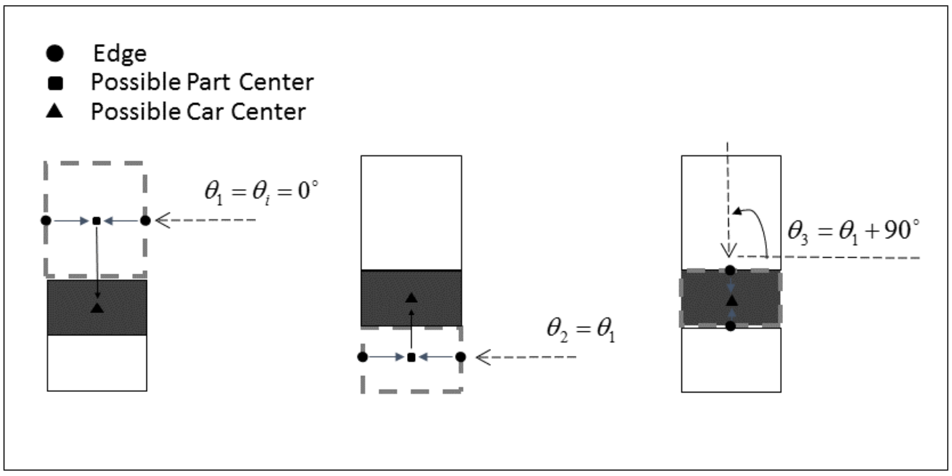
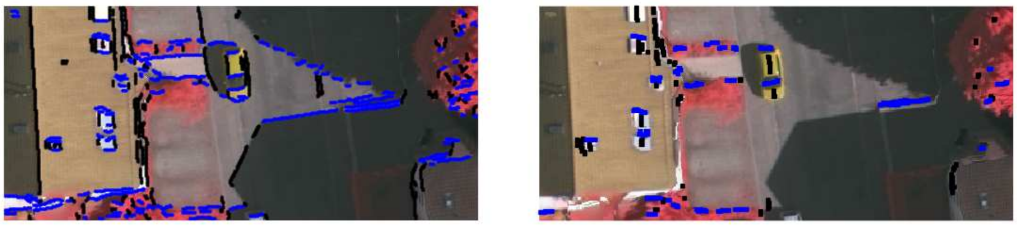


Figure 4.14. The relative geometry between car parts and voting locations.

vary, however the width of the car is assumed to be in a certain interval such as between 1 m and 2.5 m. Thus, we only try to find the ribbons with this width. As shown in the first part of Figure 4.14, the center of the ribbon (black square marker) is extracted first. Then, this ribbon center votes for the windshield location, towards $\theta_1 + 90^o$. In the second step, the front end rectangle votes towards the windshield location, in the direction of $\theta_1 - 90^o$. At this point, since we don't know which rectangle belongs to which part of the car, they will both vote towards south and north directions. At the third step, we find dark rectangles (windshield) where it is already assumed to be the car center. We have observed that most of the times one of the two sides of the windshield could not be detected by Canny edges due to color of the road and the car. So, here we try to find a dark rectangle in the direction of $\theta_3 = \theta_1 + 90^o$.

We show our approach step by step on the sample image. First we extract directional edges. In Figure 4.16.a the extracted edges for $\theta_1 = \theta_2 = 0^o$ and $\theta_3 = 90^o$ are given with black and blue dot markers respectively. The extracted ribbon centers are given in Figure 4.16.a with black and blue markers. The black markers represent possible front end (R_1) or back end (R_2) rectangles and blue markers represent possible windshield centers (R_3). The ribbon center locations are denoted as $(\hat{x_i}, \hat{y_i})$.

As we mentioned previously, each part of the car will vote for possible windshield location. The front end rectangle (R_1) votes for the possible windshield location, $(\bar{x_i}, \bar{y_i})$ as



a. Directional edges. b. Rectangle centers.

Figure 4.15. Edges and rectangle centers for $\theta = 0$.

$$\bar{x_i} = \hat{x_i} + r\cos(\theta_1 - 90^0)$$
$$\bar{y_i} = \hat{y_i} + r\sin(\theta_1 - 90^0) \tag{4.22}$$

In this equation, r is the voting distance. There are multiple rectangle centers side by side. So, there is a good chance that at least one of the rectangle center point will vote for the exact windshield center. Also the size of the cars do not change extremely. We also use a probabilistic voting scheme thus the uncertainty of the voting distance is expected to be handled by the method. Thus, we choose the voting distance r as a constant parameter.

The back end rectangle (R_2) voting location is in the opposite direction as

$$\bar{x_j} = \hat{x_j} + r\cos(\theta_1 + 90^0)$$
$$\bar{y_j} = \hat{y_j} + r\sin(\theta_1 + 90^0) \tag{4.23}$$

where $(\hat{x_j}, \hat{y_j})$ are the possible back end rectangle centers. The windshield rectangles (R_3)

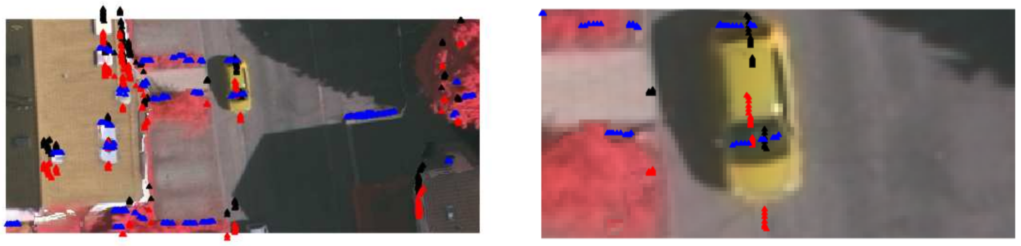

a. Vote locations. b. Vote Locations, zoomed on a car

Figure 4.16. Vote locations for $\theta = 0^o$.

voting location is the rectangle center itself.

$$\bar{x_k} = \hat{x_k}$$
$$\bar{y_k} = \hat{y_k} \tag{4.24}$$

where $(\hat{x_k}, \hat{y_k})$ are the possible windshield rectangles (R_3) center locations. In Figure 4.16 we give the voting locations. Each different color votes belong one of three rectangles. As it is seen in Figure 4.16.b the three rectangles (R_1, R_2, R_3) votes are gathered on the windshield.

Since we assumed the windshield is the possible car center and trying to vote for only possible windshield center points, we weight each vote with the gray color of the image as

$$w_i = exp\left(-\left(\frac{I(\bar{x_i}, \bar{y_i}) - a}{b}\right)^c\right) \tag{4.25}$$

where $I(\bar{x_i}, \bar{y_i})$ is the gray color of possible windshield center. We have observed that the gray color of windshields are mostly below 100. For $a = 30$, $b = 80$ and $c = 6$ the weight

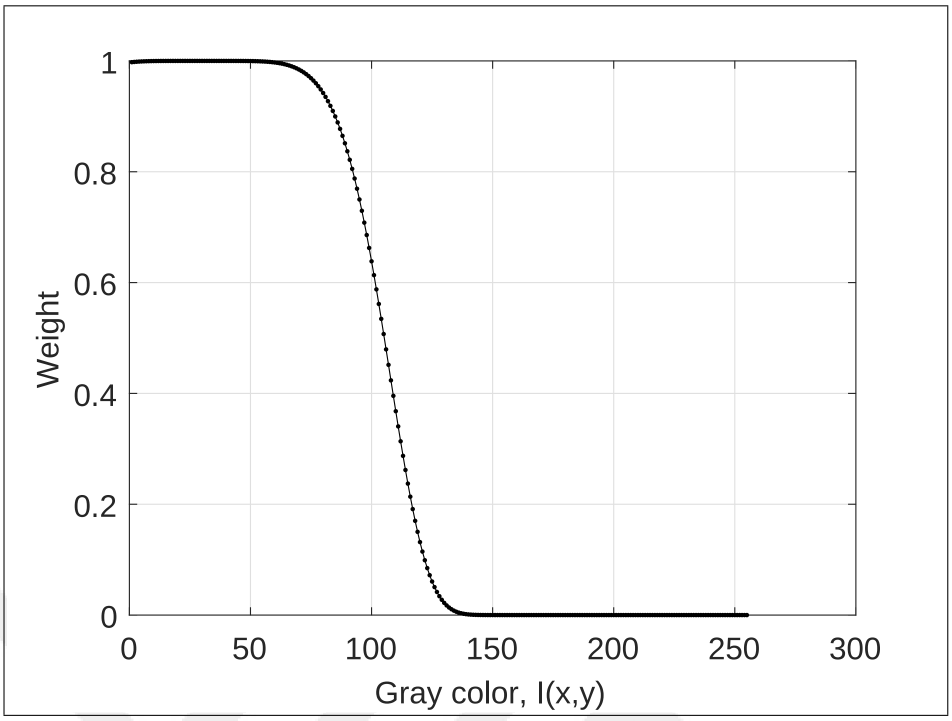


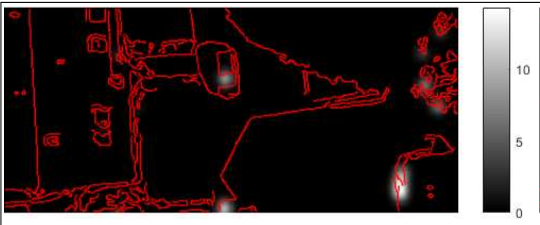
Figure 4.17. The weight function for voting car center.

function response is plotted as in Figure 4.17.

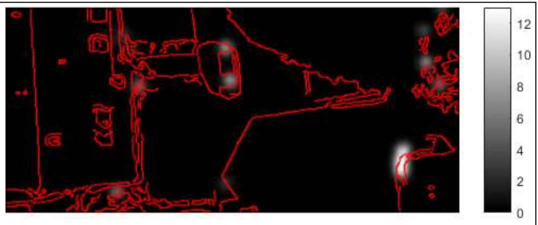
We use a joint probabilistic model to locate the center of the car. Given three rectangles, the pdf of locations of possible car centers (windshields) are modeled as

$$p_{C,\theta_i}(x, y|R_1) = \sum_{i=1}^{N_i} w_i \exp\left(-\frac{(x-\bar{x_i})^2 + (y-\bar{y_i})^2}{2\sigma_i}\right) \tag{4.26}$$

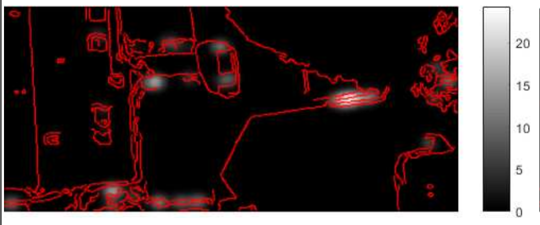
$$p_{C,\theta_i}(x, y|R_2) = \sum_{j=1}^{N_j} w_j \exp\left(-\frac{(x-\bar{x_j})^2 + (y-\bar{y_j})^2}{2\sigma_j}\right) \tag{4.27}$$



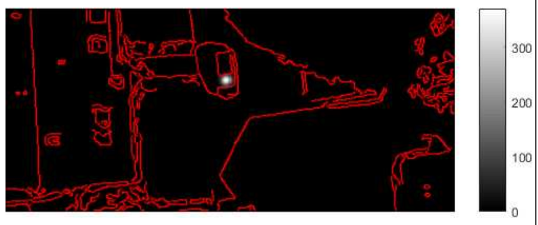
a. $p_{C,\theta_i}(x,y|R_1)$



b. $p_{C,\theta_i}(x,y|R_2)$



c. $p_{C,\theta_i}(x,y|R_3)$



d. $p_{C,\theta_i}(x,y|R_1,R_2,R_3)$

Figure 4.18. Car center pdf estimates for $\theta_i = 0^0$

$$p_{C,\theta_i}(x,y|R_3) = \sum_{k=1}^{N_k} w_k \exp\left(-\frac{(x-\bar{x_k})^2 + (y-\bar{y_k})^2}{2\sigma_k}\right) \tag{4.28}$$

We assume each of the rectangle as a feature and thus for a car in the direction of θ_i the car centers are modeled as the joint multiplication of the pdfs.

$$p_{C,\theta_i}(x,y|R_1,R_2,R_3) \propto p_{C,\theta_i}(x,y|R_1)p_{C,\theta_i}(x,y|R_2)p_{C,\theta_i}(x,y|R_3) \tag{4.29}$$

In Figure 4.18 we plot the pdf for each of the three rectangles and the final estimated pdf when $\theta_i = 0^o$. On the windshield of the car the relative probability is much higher than any of other locations.

Here we assumed the car is in a specific direction. We need to find the cars with all possible orientations. So we marginalize the pdf in Eqn. 4.29 for $\theta \epsilon [0, \pi/N, 2\pi/N, ..., (N-1)\pi/N]$ and estimate the final pdf for all cars as follows.

$$p_C(x, y|R_1, R_2, R_3) \propto \sum_{i=1}^{N} p_{C,\theta_i}(x, y|R_1, R_2, R_3) \quad (4.30)$$

Finally, we will find the modes of the final estimated pdf and assume the modes are possible car centers.

4.4. EXPERIMENTS ON OBJECT DETECTION

In previous sections, we explained our method for the detection of various objects centers in remote sensing data. In this section, we will show the experiment results of object detection. We tested the proposed method on each object type. We used different feature sets for different objects. In all of the measurements, we take the object based performance measures. As a result, we count the number of objects in the test images and obtain the object based detection performances.

We will give detection results on sample images for each object type. For a fixed detection threshold, we will summarize the performance of the method for all images in terms of True Positive (TP), False Positive (FP), and False Negative (FN) metrics. If the local maxima of estimated pdf are anywhere on the object pixels, then we count these as TP. If there are detections on non-object pixels, then we count them as FP. If there are multiple detections on the same object, we also count these as FP. If we can not find any detection for an object, then we count this as FN.

We will also give precision (Correctness - CR), recall (Completeness - CP), and quality (Q) metrics for accuracy assessment. These metrics are heavily used in computer vision applications to see how well the algorithm is performing.

These values can be obtained as follows.

$$
\begin{aligned}
CR &= \frac{TP}{TP+FP} \\
CP &= \frac{TP}{TP+FN} \qquad (4.31) \\
Q &= \frac{TP}{TP+FP+FN}
\end{aligned}
$$

Using precision and recall metrics, we will make up the PR (Precision and Recall) curve [116]. Hence, we will see the performance of the method when the detection threshold is varied. PR curve shows the performance of the method when a parameter in the method is varied. We would like to see the algorithm's performance at the top-left corner of the PR curve.

4.4.1. Building Detection Experiments

Here, we test our method on panchromatic images (2D) and DSM data (3D). So, in the following two subsections, we analyze the results separately.

4.4.1.1. Experiments with 2D Data

We tested our method on Ikonos satellite images. The satellite images area acquired over Adana, Turkey. There are total number 23 image sets with a total number of 674 buildings. Some of the images are given in Figure 4.19. The spatial resolution of the images is 1 m. We think this large data set represents diverse building types. We manually extracted the pixels for each building individually. For a fixed detection threshold, we will summarize the performance of the method.

We use five features in the building detection experiments. These are edge point pairs feature, edge L-shape feature, steerable filter feature, Harris corner feature, and shadow respectively.

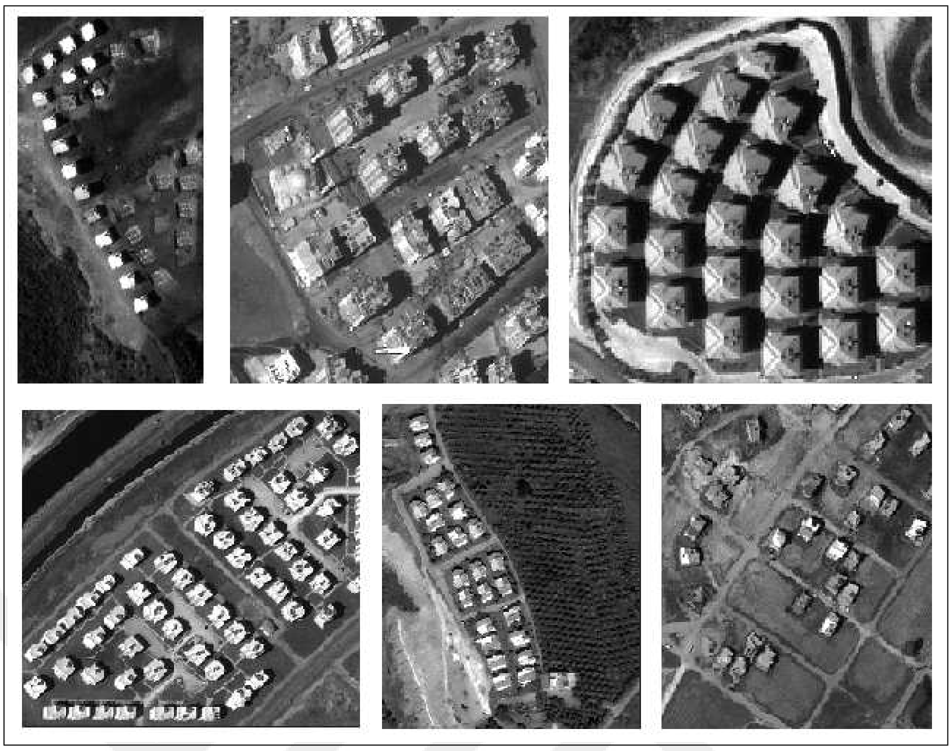


Figure 4.19. Samples for Adana satellite images.

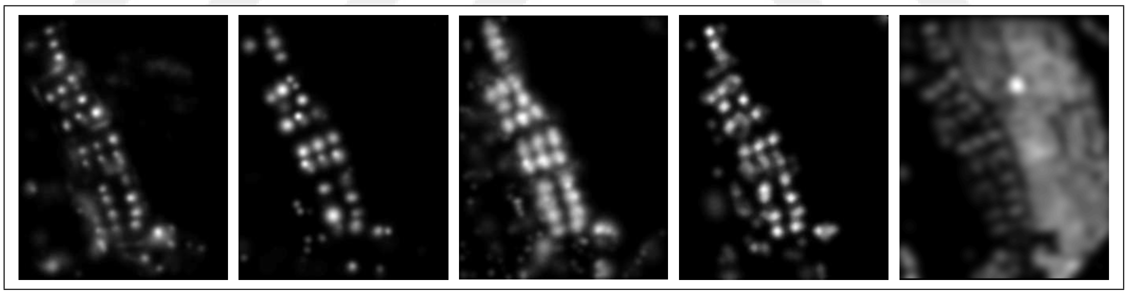


Figure 4.20. Estimated pdf of buildings for each feature for Adana7 test image.

On a single test image, we will show the estimated pdfs and detection results. In Figure 4.21.b, the Adana7 image is given. On this image, we extracted five features and estimated building pdfs for each individually as described in the previous sections.

In Figure 4.20, we give the estimated building pdfs for each extracted features. In this figure, we used the following features from left to right: edge point pairs feature, edge L-shape feature, steerable filter feature, Harris corner feature and shadow. The lower and higher probabilities are shown with dark and light colors, respectively. On this pdfs, the probability

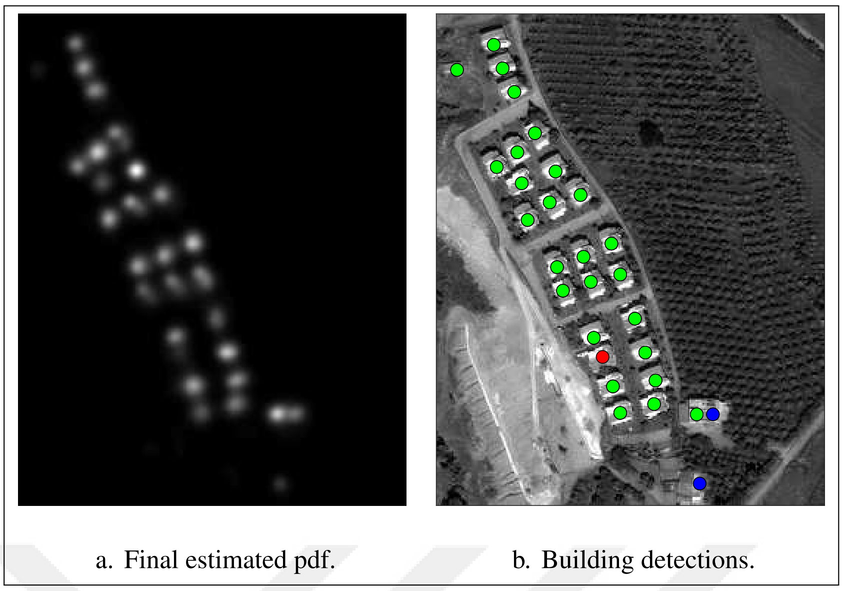


Figure 4.21. Final estimated pdf using all five features and building detection results.

at building centers are high. There are also some higher probabilities on non-building spots. For instance, if only shadow information has been used, there would be many false alarm as the trees would also marked as buildings. However, when all information is used together, we expect to see high probabilities on building centers only. In Figure 4.21.a, we give the estimated pdf when all five features are used. All of the high probabilities on non-building spots are decreased.

In Figure 4.21.b, we give the building detection results where green, blue and red markers indicate true detection, false alarm and miss detection respectively.. On the final estimated pdf, we find the local maximums and assume these are the building centers. While doing this, we use a threshold such that if there is any local maximum on the pdf has a probability lower than this threshold, then we discard it. Since the final pdf is merely the product of five individual pdfs, we would see low probabilities at the center of buildings. For example, assume that for each five pdfs the probability of a pixel being the building center is 0.6. Then in the final pdf, this pixel would have a probability $0.6^5 = 0.0778$. So even though for one pdf the probability of being the building center is high, in the final pdf we see a much lower probability for that pixel. For this reason, we use a threshold as low as 0.001 to detect the buildings.

In Adana7 image, there are 27 buildings where we detected 26 of them correctly. There is only one FN and two FP. In Table 4.1 we give the results for 23 Adana images. There are 674 building in total. In this table we give the results when the features are used individually, in combinations and all together. Even though the TP numbers are really high when one feature is used, FP numbers are also high. When all features are used, the FP numbers are decreased dramatically. Out of 674 buildings, we correctly detected 631 of them, there are 62 false alarms and 43 missed detections.

Table 4.1. Building detection results for Adana images.

Feature	**Building**	**TP**	**FP**	**FN**	**CR**	**CP**
f_1	674	651	618	23	0.513	0.966
f_2	674	617	305	57	0.669	0.915
f_3	674	652	871	22	0.428	0.967
f_4	674	618	566	56	0.522	0.917
f_1, f_2	674	654	212	20	0.755	0.970
f_1, f_2, f_3	674	657	149	17	0.815	0.975
f_1, f_2, f_3, f_4	674	628	94	46	0.870	0.932
f_1, f_2, f_3, f_4, f_5	674	631	62	43	0.911	0.936

We varied the detection threshold from $1e - 7$ to 0.5 and measured the performance of the method. The results are plotted in Figure 4.22. The result that is closest to top-left corner on the PR curve is the best result which is the case when detection threshold is 0.001. When all features are used together, FP numbers are decreased dramatically and thus the precision of the algorithm is good.

4.4.1.2. Experiments with 3D Data

In this section, we test our method on DSM data. We used the ISPRS data set which contains 33 patches having different size [117]. In each patch, there exists a DSM and true orthophoto (TOP) image. Spatial resolution of the data set is 9 cm. We have used 8 images of the data set. All of the images are given in Figure 4.23. Here the

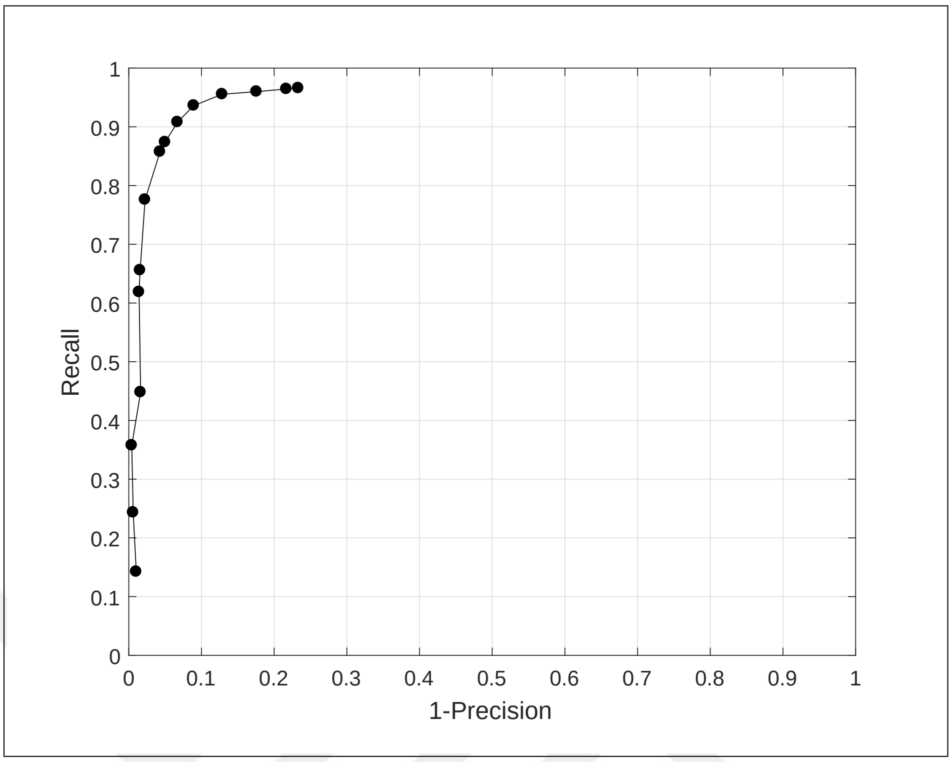


Figure 4.22. PR curve for 23 Adana test images with 674 buildings in total.

top row image names are $top\_mosaic\_09cm\_area1\_3\_5\_7$ boottom row image names are $top\_mosaic\_09cm\_area9\_11\_13\_15$.

In this section, we extract the features using only DSM data. We have extracted the edge point pairs, edge L-shape feature, steerable filter feature, Harris corner detector feature and DSM height information. We have extracted the the pdf of building centers for each of the feature. We illustrate the results on $top\_mosaic\_09cm\_area3$ image where the orthophoto image and DSM is given in Figure 4.25. In Figure 4.24 we give the estimated building pdfs for each future.

Using Eqn. 4.11, we plot the final pdf of buildings $p_B(x, y|f_1, f_2, f_4, f_5, f_7)$ in Figure 4.25.c. In these results we used the following features from left to right: edge point pairs feature, edge L-shape feature, steerable filter feature, Harris corner detector feature and height (DSM)As can be seen in the figure, the pdf estimation has high probability at most of the building centers. As explained in the previous section, we extract the local maxima and assume these

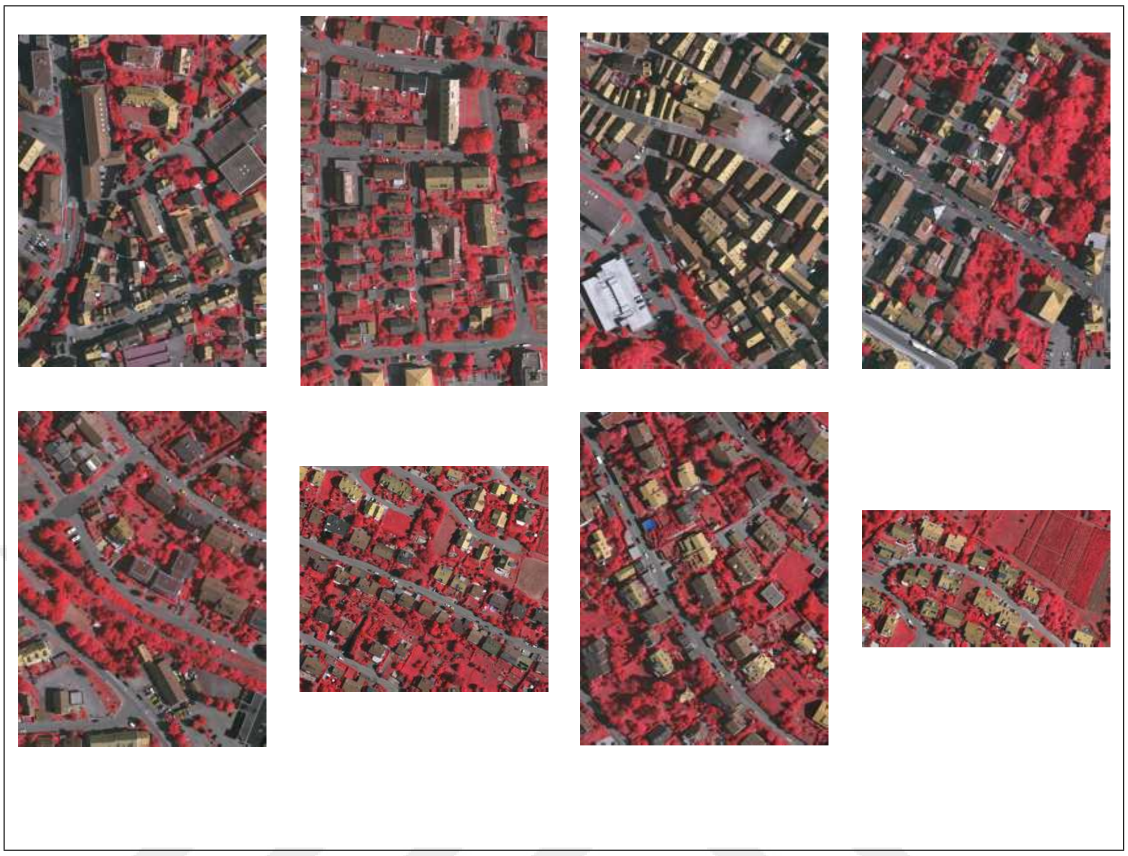


Figure 4.23. Test images of Vaihingen, Germany.

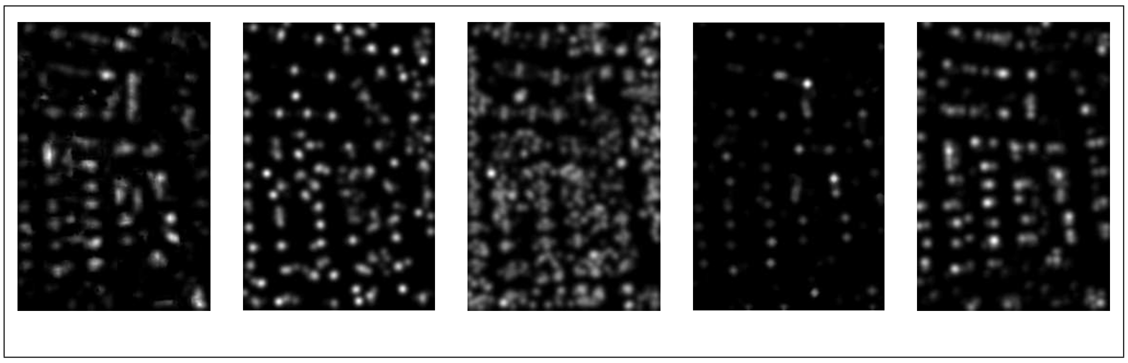


Figure 4.24. Estimated pdf of buildings for each future for $top\_mosaic\_09cm\_area3$.

are the building center locations. For each local maximum, we only accept it as a possible detection if pdf estimation at this location is higher than a threshold. The pdf in Figure 4.25.c is normalized between 0 and 1. In this example the threshold is set to $1e - 3$. In this case, the TP are labeled with green markers, FP are labeled with red markers and FN are labeled with blue markers in Figure 4.25.a. In this image, there are 63 buildings and we have correctly

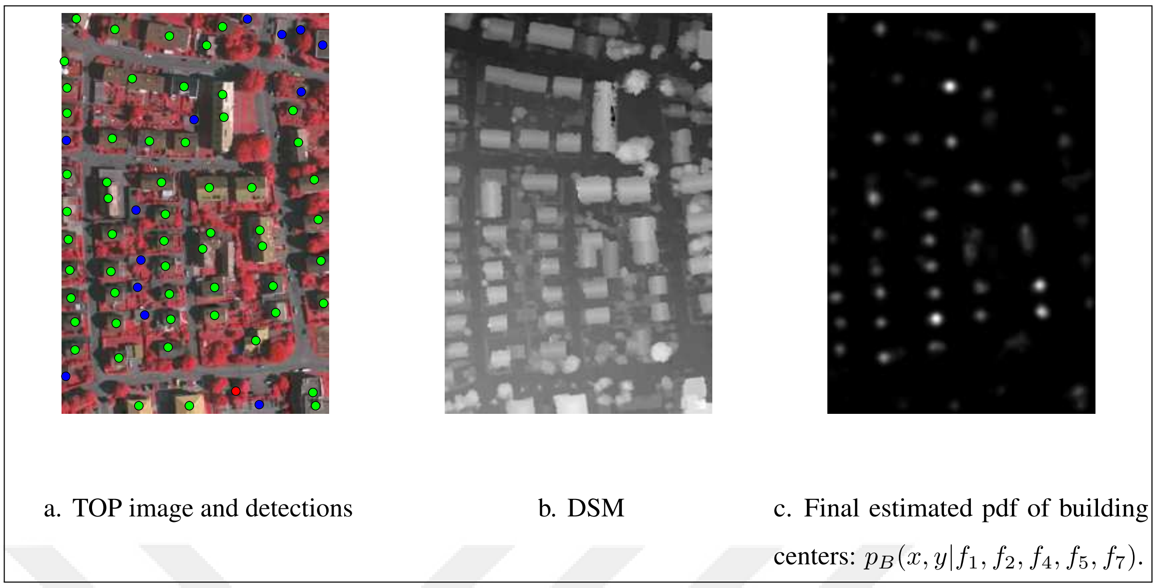

a. TOP image and detections
b. DSM
c. Final estimated pdf of building centers: $p_B(x, y|f_1, f_2, f_4, f_5, f_7)$.

Figure 4.25. Final estimated pdf using all five features and building detection results.

detected 50 of them correctly with 1 false alarm. We assume we detected the building correctly if the local maxima location are on anywhere on the building.

We tested our method on 8 images which were given in Figure 4.23. In these 8 images, there are a total of 323 buildings. When the detection threshold is set to $1e - 3$, we tabulate the object based detection results in Table 4.2.

Table 4.2. Building detection results for ISPRS Vaihingen data set.

Feature	**Building**	**TP**	**FP**	**FN**	**Precision**	**Recall**
f_1	323	281	88	42	0.76	0.87
f_2	323	235	177	88	0.57	0.73
f_4	323	259	150	64	0.63	0.80
f_5	323	222	99	101	0.69	0.69
f_7	323	281	84	42	0.77	0.87
f_1, f_2, f_4, f_5, f_7	323	269	29	54	0.90	0.83

It is shown that when the features are used individually, the TP measurements are high but

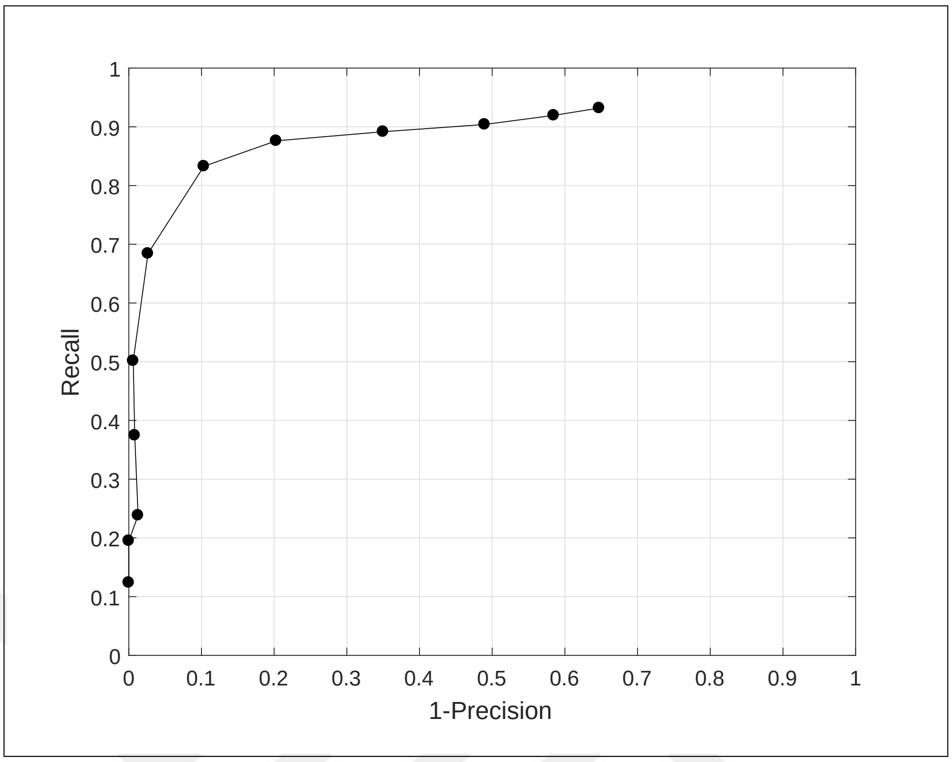


Figure 4.26. PR curve for 8 ISPRS Vaihingen images with 323 buildings.

also FP measurements are high. When all the information is merged in the final pdf, the FP measurement drops dramatically to 29 but the TP number stays at 269. Even though the recall value 0.83 is smaller then f_1 and f_7 features recall value, the precision measurement is the highest with 0.90 when all of the features are merged.

To measure the sensitivity of the method when the detection threshold changes, we calculated precision and recall values for 12 different thresholds where the smallest threshold is $1e-8$ and the largest is 0.5. We draw the PR curve in Figure 4.26. For the thresholds larger than $1e-3$, the recall values are below 0.7. The best performance is the one with the closes point to the north west part of the figure where we tabulate the results in Table 4.2. For the smallest threshold, the recall value goes up to 0.95. However in this case there are lots of false alarms also. We have shown that using a reasonable detection threshold gives very good detection results on DSM data.

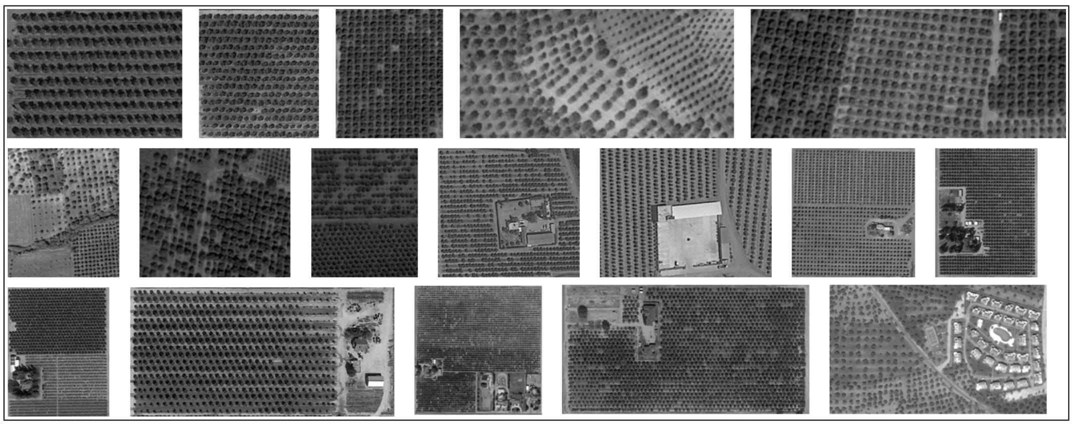


Figure 4.27. Test images used in tree detection experiments.

4.4.2. Tree Detection Experiments

We test the proposed method on 17 orchard satellite images acquired from Google Earth as in Figure 4.27. In these test images, there are a total of 13476 trees (each having a diameter between 2 to 20 pixels). Test images contain olive, peach, pear, and citrus trees. Therefore, tree type variability is satisfied.

We provide metadata for test images in Table 4.3. This metadata clearly indicates that several regions from different sides of the world are picked in testing. Therefore, the soil type and background color is not constant. This metadata may be of help for future tree detection studies as well.

We used the following parameter values throughout experiments. The range interval for neighbor search in probabilistic voting is set as $r_{min} = 1$ and $r_{max} = 25$ pixels. The rationale here is as follows. Trees in test images have diameter varying between 2 to 20 pixels. In order to handle the smallest and largest tree in neighbor search step, r_{min} and r_{max} are set accordingly. The orientation tolerance in Eqn. 3.15 is set as $\alpha = \pi/12$ rad. Setting a smaller α value may lead to missing neighbor points. A larger α value may result in sparse votes. Therefore, $\alpha = \pi/12$ rad seems to be a good choice.

In order to test the performance of the proposed method, we manually extracted tree crowns of all trees in test images. Based on these, we obtain the true positive (TP), false positive (FP), and false negative (FN) values. Here, we take object based performance measures. Hence, a tree is assumed to be detected when the ellipse representing it overlaps with the ground truth data by more than 80 %. Based on this assumption, we provide the total TP, FP, and FN values obtained from 17 test images with a total of 13476 trees in Table 4.4. As can be seen in this table, the Canny edge detector has the highest TP value with the lowest FP and FN values.

We next calculate the completeness (CP), correctness (CR), and quality (Q) metrics for accuracy assessment. Based on the TP, FP, and FN values in Table 4.4, we obtain CP, CR, and Q values for the proposed method as in Table 4.5. As can be seen in this table, the Canny edge detector provides the best performance results. Therefore, it is selected throughout the experiments. To note here, the performance obtained when different edge detectors are used

Table 4.3. Metadata for test images of orchard trees.

Test Image	**Latitude**	**Longitude**	**Location**	**Image Size (pixels)**	**No. of Trees**	**Resolution (cm)**
Image 1	37^o 52’ 56.70” N	120^o 51’ 41.07” W	California, USA	324 × 238	197	30
Image 2	37^o 33’ 09.26” N	120^o 59’ 47.57” W	California, USA	251 × 270	270	30
Image 3	36^o 59’ 37.20” N	35^o 00’ 26.44” E	Mersin, Turkey	195 × 234	288	40
Image 4	39^o 08’ 25.44” N	27^o 46’ 48.82” E	Manisa, Turkey	386 × 181	354	40
Image 5	36^o 56’ 00.74” N	35^o 00’ 17.17” E	Mersin, Turkey	453 × 185	425	40
Image 6	36^o 50’ 18.58” N	35^o 20’ 07.93” E	Adana, Turkey	367 × 423	380	40
Image 7	36^o 59’ 12.50” N	35^o 00’ 09.60” E	Mersin, Turkey	265 × 226	322	40
Image 8	36^o 59’ 33.76” N	35^o 00’ 24.60” E	Mersin, Turkey	278 × 336	432	40
Image 9	36^o 48’ 00.76” N	35^o 07’ 58.49” E	Mersin, Turkey	399 × 363	655	40
Image 10	36^o 47’ 09.07” N	35^o 07’ 23.02” E	Adana, Turkey	420 × 316	519	30
Image 11	36^o 45’ 16.26” N	119^o 30’ 58.78” W	California, USA	466 × 488	1193	30
Image 12	36^o 44’ 51.29” N	119^o 31’ 36.08” W	California, USA	368 × 465	1137	40
Image 13	36^o 42’ 32.16” N	119^o 35’ 52.18” W	California, USA	438 × 492	1427	30
Image 14	36^o 38’ 50.49” N	119^o 39’ 37.31” W	California, USA	665 × 324	564	30
Image 15	36^o 39’ 27.14” N	119^o 45’ 39.56” W	California, USA	500 × 506	3273	60
Image 16	36^o 37’ 13.22” N	119^o 50’ 31.74” W	California, USA	756 × 394	1555	40
Image 17	39^o 25’ 33.63” N	26^o 49’ 00.06” E	Balikesir, Turkey	493 × 292	485	50

is also close to the Canny edge detector. Therefore, we can claim that the proposed method does not heavily depend on the edge detector type in operation.

We next compare the proposed method with the ones in literature. Therefore, we first pick the local maximum filtering method [36]. Then, we pick Ozdarici-Ok's method for comparison [58].

In order to compare the proposed method with local maximum (LM) filtering, we implemented the LM filtering. To obtain a good result from LM filtering, we had to smooth the image with a Gaussian kernel with different widths. Then for each width, we separately extracted the local maxima of the filtered image for detecting tree crowns. The best result is obtained when the width of Gaussian kernel is set as three pixels.

Table 4.4. TP, FP, and FN values using different edge detectors.

Edge detector	**TP**	**FP**	**FN**
Sobel	11827	2531	1649
Prewitt	11867	2585	1609
Roberts	11299	2740	2177
LoG	11625	2406	1851
Canny	12141	2065	1335

Table 4.5. CP, CR, and Q values in percentages using different edge detectors.

Edge detector	**CP (%)**	**CR (%)**	**Q (%)**
Sobel	80.55	89.81	74.93
Prewitt	80.60	90.20	75.17
Roberts	78.96	84.27	70.17
LoG	82.32	91.08	76.39
Canny	84.73	93.94	80.51

As can be seen in Table 4.5, we obtain CP, CR, and Q values for the proposed method as **84.73** %, **93.94** %, and **80.51** % respectively. These values were 89.07 %, 83.80 %, and 76.15 % for the LM filtering method. Comparing these, it can be seen that the proposed method has better CR and Q values with less CP value. In terms of the CR metric, the proposed method has almost 10 % improvement compared to the LM filtering method. One possible explanation of this improvement is due to the shadows in the image. If there are heavy shadows, then the LM filtering method is negatively affected from it. The other reason may be using a single filter size in operation which may not be suitable for small and large trees at once.

We also compare the proposed method with Ozdarici-Ok's method using their image set. To do so, we applied our method to three of their test images. We provide the test results in Table 4.6. To be consistent with their naming, we provided the results in terms of precision and recall which correspond to CR and CP values respectively. As can be seen in this table, among three test images only in one of them the proposed method has better results. Besides, the other results are similar at best. There are two explanations for this result. First, the proposed method does not depend on multispectral information. However, Ozdarici-Ok's method uses this information. Second, the proposed method does not assume a specific tree type. Whereas Ozdarici-Ok's method focuses on citrus trees. Therefore, the proposed method could not perform as good as Ozdarici-Ok's method in their test set.

Table 4.6. Comparison of the proposed method with Ozdarici-Ok's method.

	Ozdarici-Ok		**Proposed**	
Image	**Precision**	**Recall**	**Precision**	**Recall**
Image I	95.8	91.2	77.5	90.2
Image II	68.5	66.6	71.0	36.3
Image III	86.2	79.8	80.1	67.3

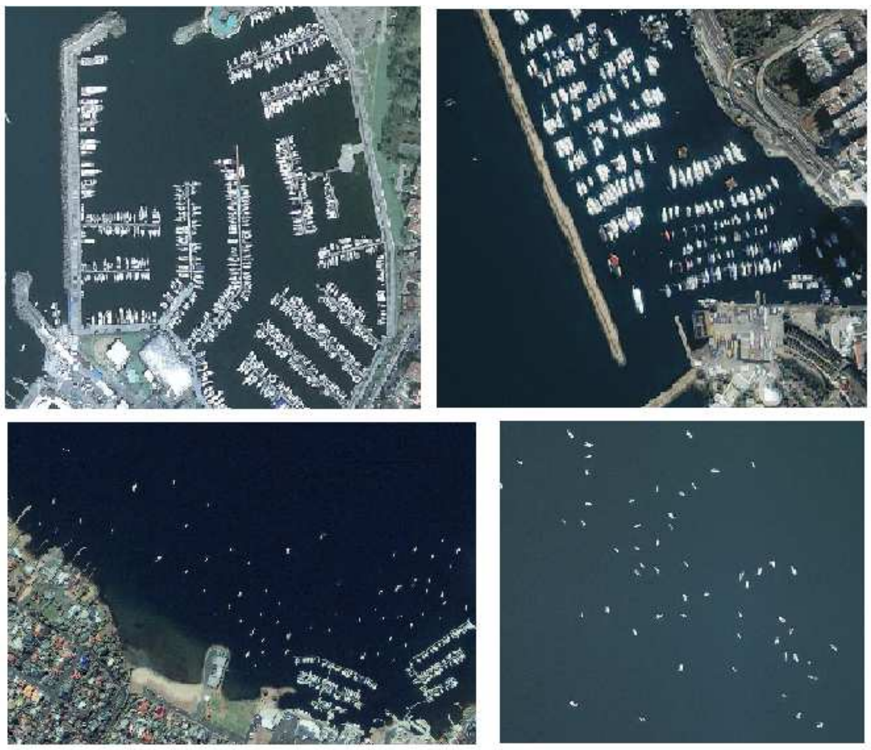
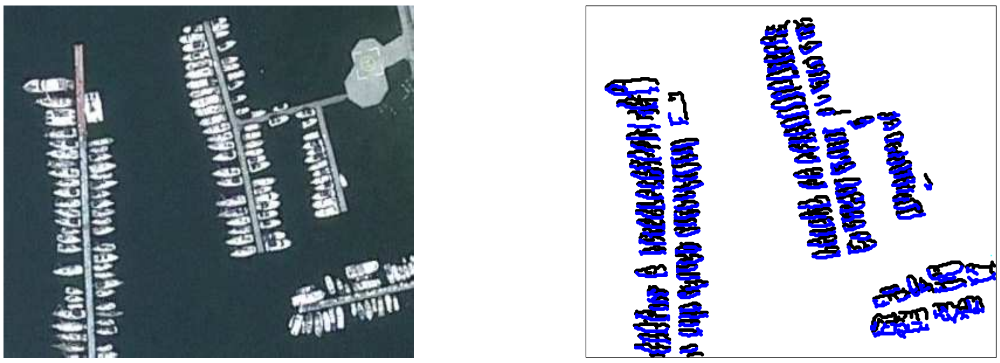


Figure 4.28. Satellite images of ships.

4.4.3. Ship Detection Experiments

Detecting and locating ships in satellite images has been extensively studied in the literature. Majority of the studies used SAR images for this purpose. Recently, researchers proposed methods based on optical satellite images. Here we test our method using optical satellite images. We tested our method on four satellite images. These images are given in Figure 4.28.

We aim to detect both open sea and inshore images. Ships are mostly in ellipsoidal shapes. We have observed that when the steerable filter is applied perpendicular to the ships orientation, we obtain high response on the ship's right and left sides. In Figure 4.29.a we give a Geoeye satellite image of ships in a harbor. Here the ships are so closely located. When we apply steerable filter along the y-axis we get high positive response at the upper side of the ships.



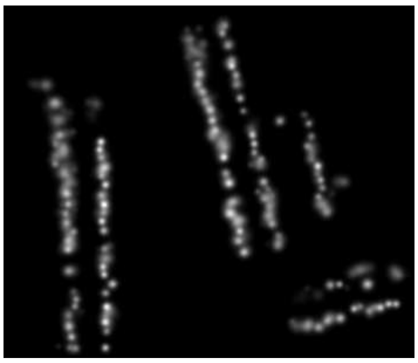
a. Sample image. b. Edges from steerable filter when $\theta = 90^o$

Figure 4.29. Sample image and edges from steerable filter result.

We also see high negative response at the lower side of the ships. In Figure 4.29.b we give the thresholded results of these negative and positive responses with black and blue dots respectively.

After this, we paired these responses as explained in Section 3.2.3 and give a vote to the center of the positive and negative responses for pdf estimation of possible ship centers. After doing this for multiple orientations we estimate the final pdf of ship centers which is given in Figure 4.30.a. The TP detections are given with green, FP detections are given with red and FN detections are given with blue markers.

In Table 4.7, we summarize the results of ship detection. We used three different satellite's images; Geoeye, Ikonos and Quickbird. Geoeye panchromatic images have spatial resolution of 0.41 m. The spatial resolution of Ikonos and Qickbird panchromatic images are 1 m and 0.61 m respectively. Hence, we tested our method on different resolutions. There are a total of 1389 ships in 5 satellite images. We correctly detected 1183 images with 59 FP and 206 FN. The precision value is 0.91 and Recall value is 0.77. The difficulty of ship detection lies on the separation of touching ships in the harbor. In some cases these ships are so close that multiple ships look like a single object. Our method successfully separated most of the ships in such cases.


a. Estimated pdf of ship centers.

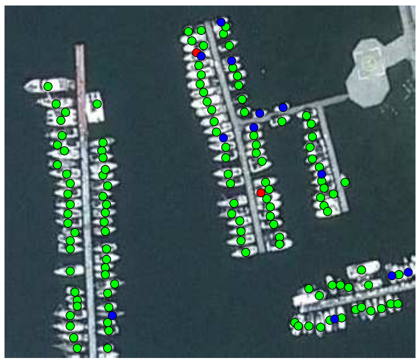

b. Detections.

Figure 4.30. Estimated pdf of ship centers and ship detections.

4.4.4. Airplane Detection Experiments

We tested the proposed method for airplane detection on 12 satellite images. These images are acquired from Google Earth and some of them are shown in Figure 4.31. There are a total of 170 airplanes in 12 images. The sizes of the airplanes vary. The illumination conditions also vary in the test images.

We provide the metadata for test Images in Table 4.8. We picked 6 different airports where

Table 4.7. Ship detection results.

Image	**Ships**	**TP**	**FP**	**FN**	**Precision**	**Recall**
Geoeye Image-1	722	641	18	81	0.97	0.89
Geoeye Image-2	56	55	2	1	0.96	0.98
Ikonos Image-1	210	167	16	43	0.91	0.80
Ikonos Image-2	238	195	10	43	0.95	0.82
Quickbird Image-1	163	125	13	38	0.91	0.77
Total	1389	1183	59	206	0.95	0.85

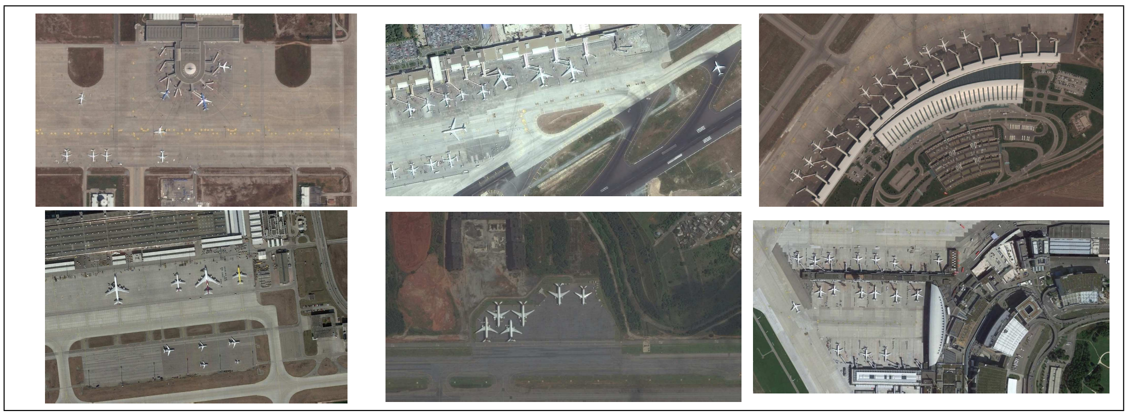


Figure 4.31. Airplane images: Top row Image 1-3-5, bottom row Image 7-9-11.

three of them are located in Turkey, one in Germany, one in Brazil and one in Switzerland. For each airport there are two different image sets. Some of these images are either collected in different times, or some of these two collected in the same time but spots the different locations of the same airport. We provided the latitude and longitude of each of the images. We also provide the number of airplanes in these images in Table 4.8.

Table 4.8. Metadata for test images of airplanes.

Test Image	**Latitude**	**Longitude**	**Airport**	**Airplane Number**
Image 1	36^0 53’ 46.59” N	30^0 48’ 03.86” E	Antalya Airport - 1	9
Image 2	36^0 53’ 46.59” N	30^0 48’ 03.86” E	Antalya Airport - 2	17
Image 3	40^0 59’ 19.26” N	28^0 49’ 59.70” E	Istanbul Ataturk Airport - 1	14
Image 4	40^0 58’ 40.41” N	28^0 49’ 40.53” E	Istanbul Ataturk Airport - 2	15
Image 5	40^0 06’ 56.78” N	32^0 59’ 30.10” E	Ankara Esenboga Airport - 1	11
Image 6	40^0 06’ 56.78” N	32^0 59’ 30.10” E	Ankara Esenboga Airport - 2	10
Image 7	48^0 20’ 53.74” N	11^0 46’ 11.20” E	Munich Airport - 1	9
Image 8	48^0 21’ 10.70” N	11^0 46’ 49.12” E	Munich Airport - 2	14
Image 9	23^0 25’ 26.24” S	46^0 27’ 35.69” W	Sao Paulo Airport - 2	6
Image 10	23^0 25’ 44.10” S	46^0 28’ 53.79” W	Sao Paulo Airport - 1	31
Image 11	47^0 27’ 09.10” N	8^0 33’ 30.62” E	Zurich Airport - 1	17
Image 12	47^0 27’ 49.32” N	8^0 33’ 17.37” E	Zurich Airport - 2	17
Total				170

In order to test the proposed method's performance, we manually extracted the pixels for each airplane in the images. Based on the airplane center detection results, we obtain the TP, FP, and FN values. Here we take object based performance measures. Hence an airplane is assumed to be detected when the methods detection result is anywhere on the airplane. We also calculate precision and recall measures. All these measures are already explained in Section 4.4.1.

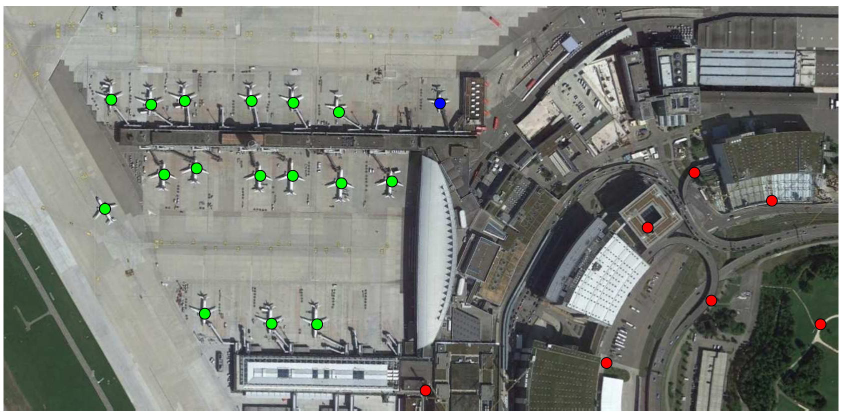
In our experiments, we only used the complex model approach. So we only benefit from the directional edges and try to extract three rectangles where these rectangle have a special relative geometry. This was already illustrated in Figure 4.7. We give pictorial results for Image 11 in Figure 4.32.

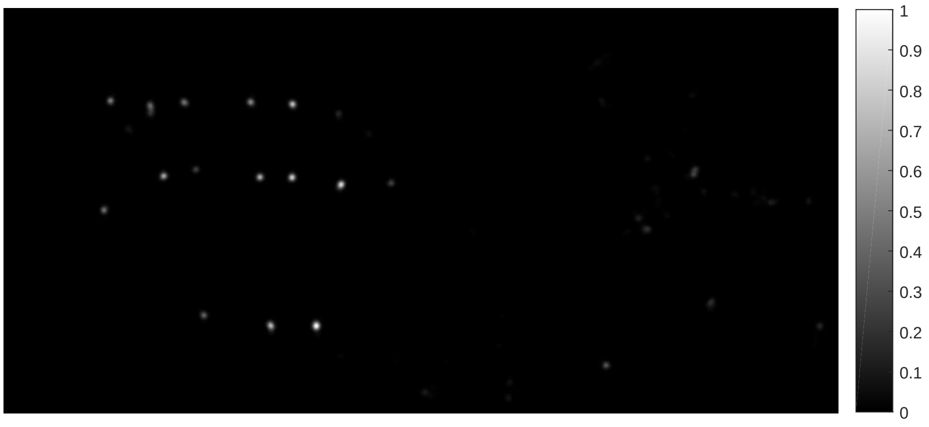
In Figure 4.32.b we give the pdf estimation of airplane center location. Here we normalized the pdf. We assume that if the local maximums of this pdf is greater than a threshold, then it is accepted as an airplane center. In this example, the threshold is set to 0.05. In Figure 4.32.a we give the detection results with colored markers. Here green, blue and red markers indicate TP, FN and FP respectively. There are 17 airplanes in the image. Here we detected 16 airplanes with 1 FN and 6 FP detections.

The false alarms are mainly located on the right side of the image where the scene is cluttered with buildings, trees and roads. Even though the relative probability of being an airplane center is low at these locations, using 0.05 threshold caused these false alarms. In Figure 4.33 we give three examples of the FP's. Notice that all of these three examples have a close geometrical similarity of an airplane (Figure 4.7).

We changed the detection threshold and calculated precision and recall values to draw the PR curve. The thresholds and detection results are summarized in Table 4.9.

We also plot the PR curve in Figure 4.34. The best results are achieved when the detection threshold is set between 0.04 and 0.1. In our approach we have only used directional edge information which is extracted from grayscale images. If the DSM data were available, then the final estimated pdf of airplane center locations would be modified with this new


a. Airplane detections.



b. Estimated pdf of airplane center locations.

Figure 4.32. Detection results for Image 11.

information and detection results would have been improved.

4.4.5. Car Detection Experiments

In our proposed car detection algorithm, we detected different parts of the car. So that we were able to separate cars from other objects. The three main part of the car, is not visible in images whose spatial resolution is below about 30 cm. For this reason, proposed car detection method requires very high resolution satellite images. Again, we benefit from the ISPRS Vaihingen data set [117]. This data set is with 9 cm spatial resolution. In this data set, DSM data is also available. We use both TOP images and DSM data.

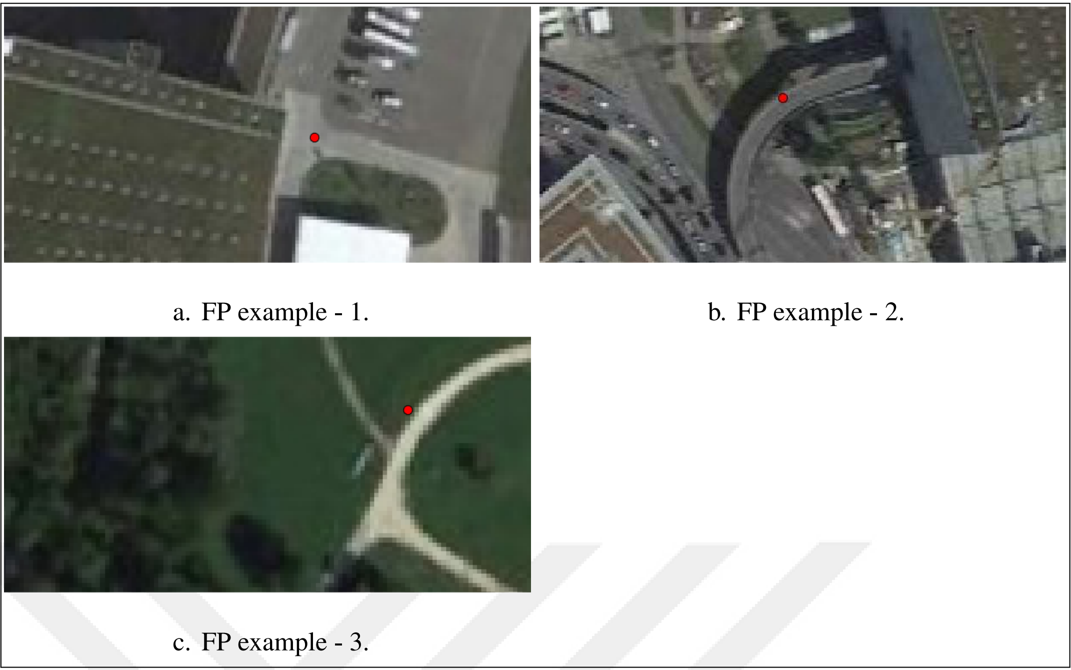


Figure 4.33. FP examples for airplane detection.

Table 4.9. Airplane detection performance for different detection thresholds.

Airplane#	**Threshold**	**TP**	**FP**	**FN**	**Precision**	**Recall**
170	0.001	160	215	10	0.427	0.941
170	0.005	158	157	12	0.502	0.929
170	0.010	156	123	14	0.559	0.918
170	0.020	155	100	15	0.608	0.912
170	0.040	152	70	18	0.685	0.894
170	0.060	149	58	21	0.720	0.876
170	0.080	146	50	24	0.745	0.859
170	0.100	142	44	28	0.763	0.835
170	0.150	129	34	41	0.791	0.759
170	0.200	121	28	49	0.812	0.712
170	0.400	86	15	84	0.851	0.506
170	0.500	67	11	103	0.859	0.394

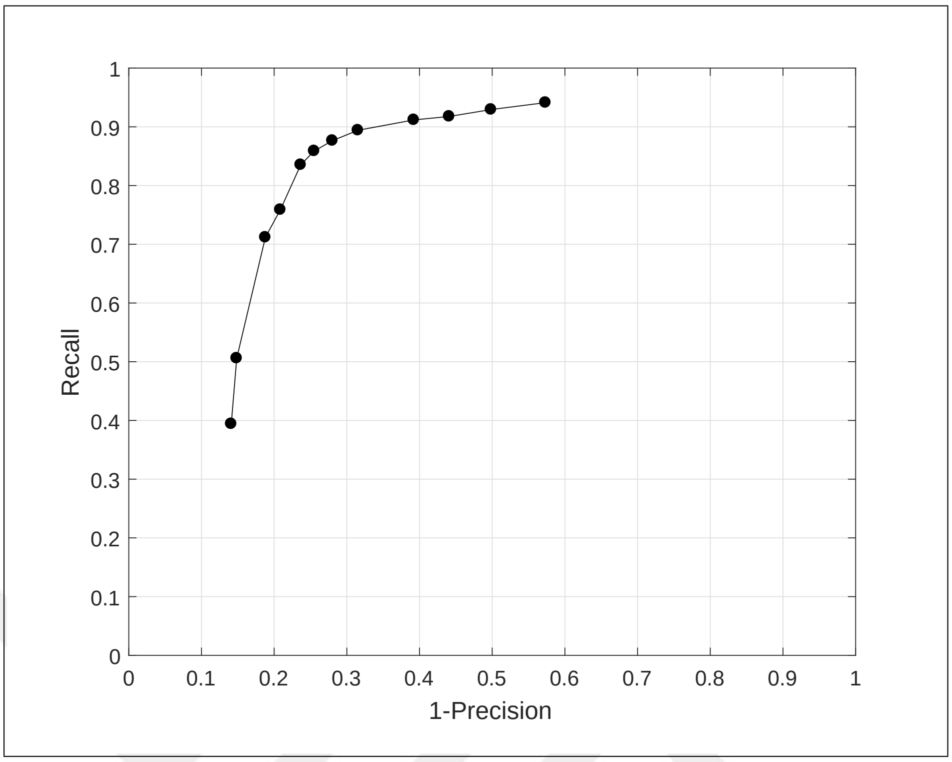
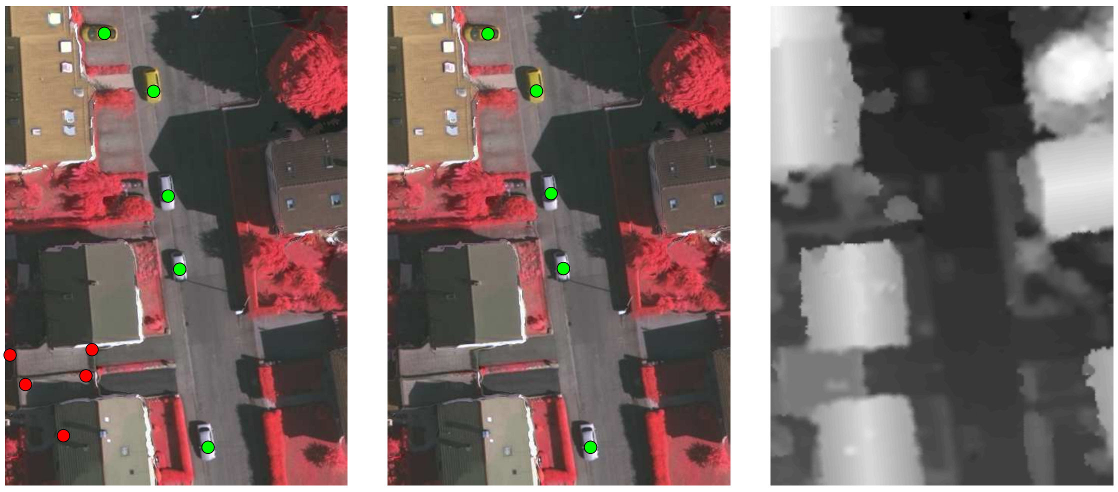


Figure 4.34. PR curve for 12 airport test images with 170 airplanes in total.

In Figure 4.35, we give car detection results on the subset of ISPRS Vaihingen $top\_mosaic\_09cm\_area3$ patch. There are 5 cars in the image. We give one result in Figure 4.35.a when only TOP image is used. In this case, we use the approach explained in Section 4.3.2 where we model a car with multiple rectangles.

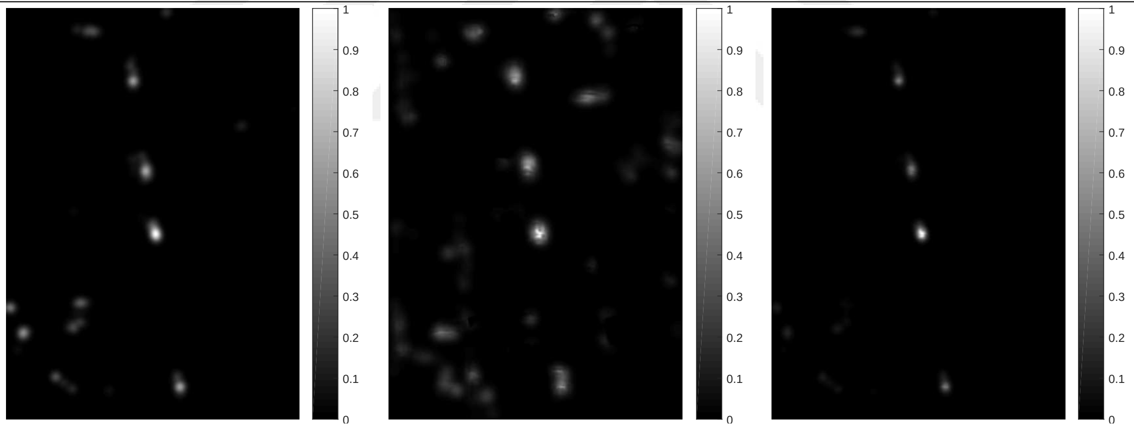
Using Eqn. 4.30, pdf estimation of car centers is given in Figure 4.36.a. We obtain high probabilities on the cars with some false alarms on the bottom left of the image. For this case the TP detections are given in Figure 4.35.a with green markers. FP detections are shown with red markers.

When we use DSM data only, we vote on cars with size and height constraints. Here an edge point votes to the highest point on DSM in a $w \times w$ neighborhood in $x - y$ directions. If the maximum and minimum height difference in $w \times w$ neighborhood for each edge point is between [0.4 m,3 m], we vote to the highest point in elevation. This way we will see high probabilities on the cars. The estimated pdf is given in Figure 4.36.b. Even though the



a. Car detections using TOP image only.
b. Car detections using TOP image and DSM together.
c. DSM of image

Figure 4.35. Car detection results on a subset of ISPRS Vaihingen patch.



a. Pdf of car centers using TOP image only.
b. Pdf of car centers using DSM only.
c. Pdf of car centers using TOP image and DSM together.

Figure 4.36. Pdf estimations of car centers.

probabilities are high on car centers, there are also false alarms on various spots. When we merge the pdfs of multiple rectangles model and height model with Eqn. 4.10 the result is given in Figure 4.36.c. We still see high probabilities on the car centers and probabilities on other spots are decreased. The detections are shown in Figure 4.35.b where we detected all cars successfully with no false alarms. Notice that all of the detection locations are on the front or rear windshield of cars as explained in Section 4.3.2.

The patch sizes of each ISPRS Vaihingen data set is large. In one patch, there are around 100 cars. The illumination conditions do not change significantly in different patches. So we only use $top\_mosaic\_09cm\_area1$ and $top\_mosaic\_09cm\_area3$ patches in our experiments. In these two patches there are 187 cars in total. We used both TOP images and DSM data. In Table 4.10, we tabulate the results when the detection threshold is varied.

Table 4.10. Proposed method's performance on the ISPRS Vaihingen data set.

Car #	**Threshold**	**TP**	**FP**	**FN**	**Precision**	**Recall**
187	1e-7	150	418	37	0.264	0.802
187	1e-6	150	375	37	0.286	0.802
187	1e-5	149	334	38	0.308	0.797
187	5e-5	149	287	38	0.342	0.797
187	0.0001	149	267	38	0.358	0.797
187	0.0003	149	222	38	0.402	0.797
187	0.0005	149	208	38	0.417	0.797
187	0.0007	149	190	38	0.440	0.797
187	0.001	149	185	38	0.446	0.797
187	0.0025	145	146	42	0.498	0.775
187	0.005	140	102	47	0.579	0.749
187	0.0075	134	78	53	0.632	0.717
187	0.01	128	63	59	0.670	0.684
187	0.05	66	19	121	0.776	0.353
187	0.1	41	9	146	0.820	0.219
187	0.2	12	2	175	0.857	0.064

In the multiple rectangles model, the final pdf is merely the product of three separate pdfs. We also use height information separately. So the detection thresholds in the final pdf should stay at low values. For this reason, we varied the detection threshold from $1e - 7$ to 0.2. In

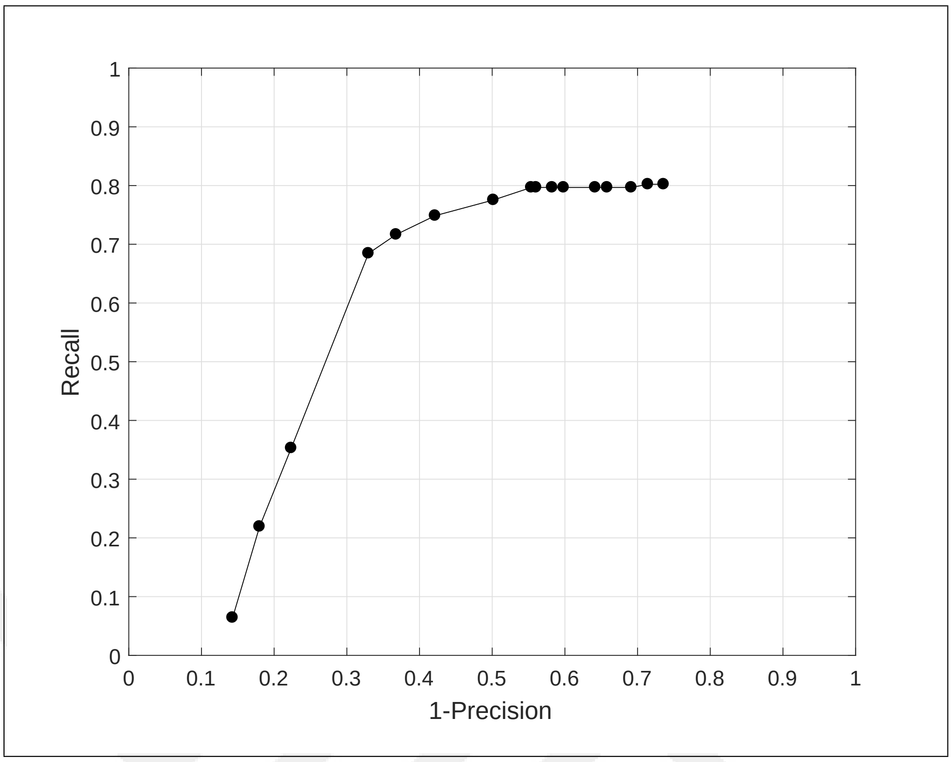


Figure 4.37. PR curve for ISPRS Vaihingen test images with 187 cars in total.

Figure 4.37, we give the PR curve. The best results are obtained when the threshold is set to 0.01 where we get 128 TP and 63 FP detections. In the next chapter, we will give the segmentation results on the same data set.

5. SEGMENTATION AND SHAPE EXTRACTION

In this chapter, we present three novel methods for segmentation and shape extraction of objects. Then, we test these with experiments on satellite images and height data. In the first section, a vote back-projection algorithm is proposed for extracting the outline of objects in satellite images in which only the gray level information is available. The second and third proposed algorithms are for segmenting height data such as DSM of objects on the ground. The second method for segmenting height data uses the local maxima of the estimated pdf of object center locations. Here, local maxima are used as seed points in segmentation. A novel segmentation method is proposed based on morphological operations.

In the experiments section, it will be shown that the proposed algorithm has certain advantages for filtering large size objects compared to other algorithms in the literature. In the third method for segmenting height data, Empirical Mode Decomposition (EMD) is used. Local, nonlinear, and non-stationary characteristics of EMD allow better DTM generation and object filtering. This method is tested on two publicly available LiDAR data set and best results in the literature are obtained.

5.1. SHAPE EXTRACTION IN 2D IMAGES

In the previous chapter, we estimated a pdf for locating object centers. In this section, we will extract the shape of objects based on the given votes in pdf estimation. In Figure 5.1 the method is demonstrated on hand drawn rectangular shapes.

Assume that we have an image and extracted the edges in it. Each edge point voted for a possible object center location. Thus, the votes are accumulated at object centers and we see high probabilities at object centers. In Figure 5.1 the z direction represents the probability of object centers and dashed lines represent the edges of objects.

Assuming these shapes belong to outlines of buildings, $P_B(x, y)$ is the probability map of building centers. At the house centers, we see local maxima of $P_B(x, y)$. At the vicinity

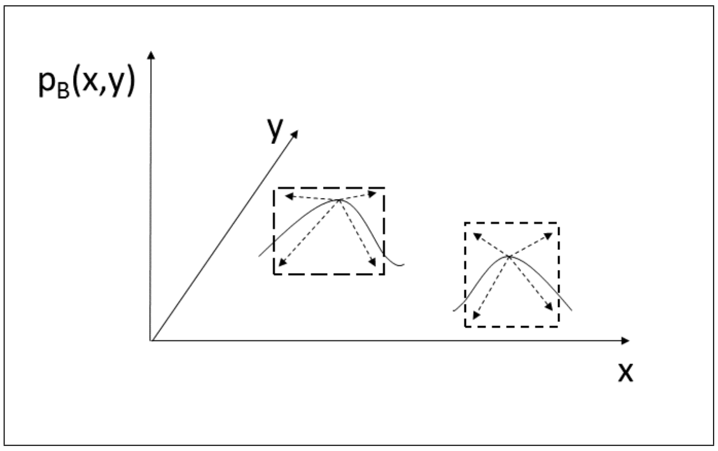


Figure 5.1. Vote back-projecting method for object outline extraction.

of these local maxima, there are given votes by building edges. We will back-project only these votes to extract each objects shape individually. In our method, we extract a shape for each of the local maximum of object center pdf. Therefore, while extracting the edges of objects, we also separate the edges for each object. Thus, we are able to label each of the object separately.

In the previous chapter, we assumed that buildings, trees and ships have simple shapes. Also, we assumed that airplanes and cars have complex shapes. Thus, we analyze extracting the shape of these objects in two separate sections.

5.1.1. Shape Extraction of Simple Objects

In Figure 5.2, we give the steps of the proposed method. Assume that we already have the pdf of object centers. Then, we can find the local maxima of this pdf and assume these are object center locations. Then, for each of the local maximum location, we can select only the most relevant votes at the vicinity of pdf local maximum location.

Based on the selected votes, we back-project the edges which voted at the vicinity of the pdf local maximum. Combining all the edge pixels, we can extract the outline of the object.

We will show the proposed method on a sample satellite image of buildings in Figure 5.3.a. In

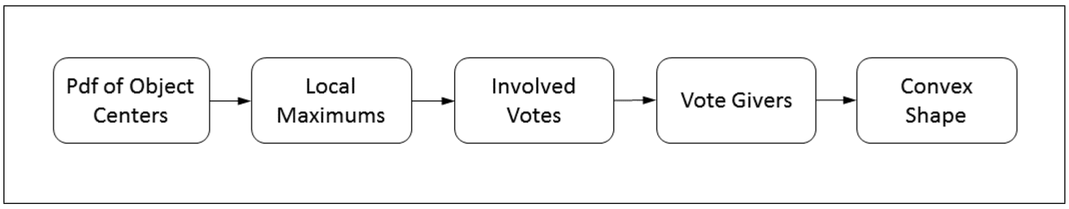
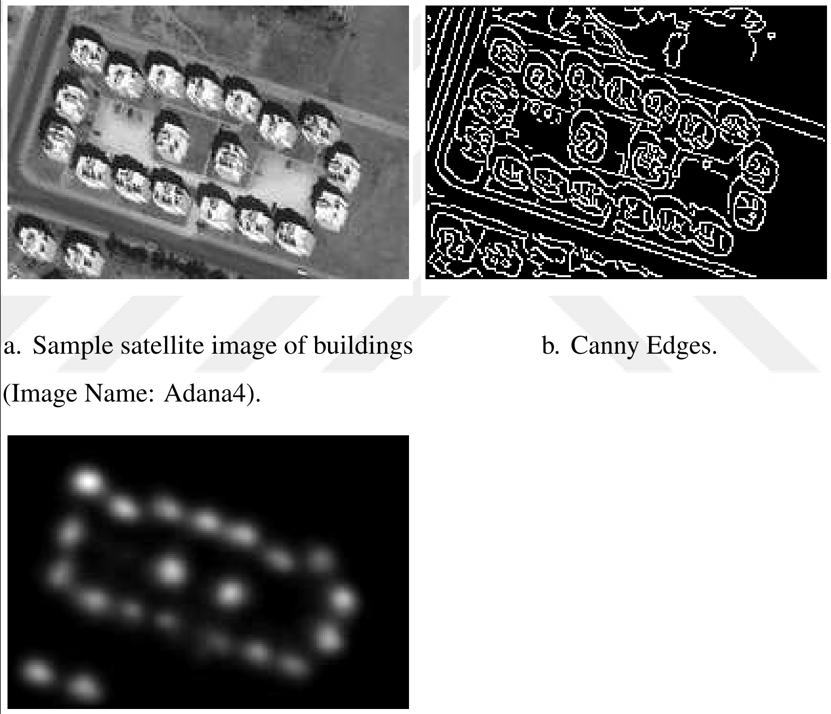


Figure 5.2. Steps of the vote back-projecting method for object outline extraction.

Figure 5.3.b, we give the Canny edges of the image. We aim to extract the edges that belong to buildings only.



a. Sample satellite image of buildings (Image Name: Adana4).

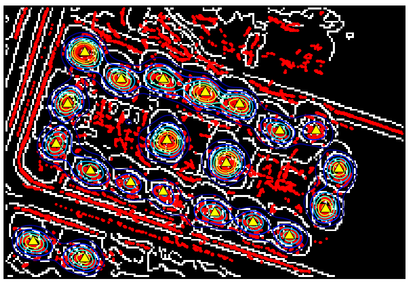
b. Canny Edges.

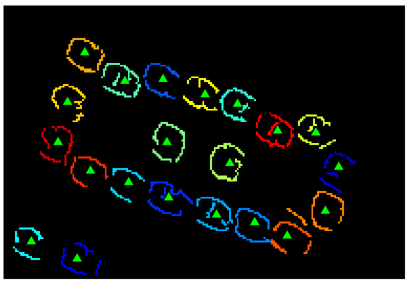
c. Estimated pdf of buildings.

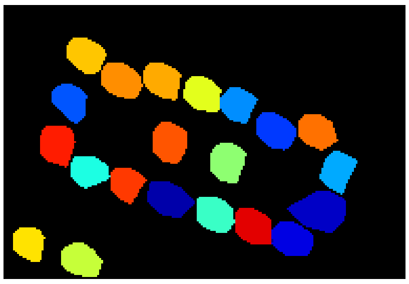
Figure 5.3. Sample satellite image of buildings, Canny edges and pdf estimation.

Assume we estimated the pdf for building center locations, $p_B(x, y|F)$, using Eqn. 4.11. Here, we used multiple features for pdf estimation such as edge based features, steerable filter based features, corners and shadows. Pdf estimation result is given in Figure 5.3.c. As expected, the probability at building centers is high. Then, we find the modes of $p_B(x, y|F)$ and take the extracted mode locations as possible building centers. Assuming we extracted K modes, then the building center locations are (x_{B_k}, y_{B_k}) for $k = 1 \cdots K$.

For each building center, we will extract the shape individually. We need to label the votes according to their relation with the possible building centers. After labeling all the votes, there will be $K + 1$ labels. This means not all votes belong to an object, some of them belong to clutters such as road segments or trees. Since we tried to use general voting rules in the previous sections, we expect to see many clutter votes.


a. Vote locations (red points) and local maximums (yellow triangles).


b. Back-projected edges of selected votes.


c. Convex shape of extracted edges.

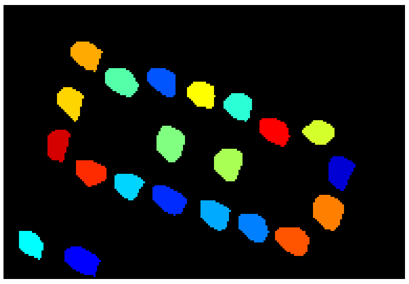

d. Chan-Vese segmentation with convex shape mask.

Figure 5.4. Back-projection and segmentation steps.

We are only back-projecting the edge based votes since they are the most suitable ones for this purpose. We have described the feature vector for an edge based ribbon as $\vec{v_1}(i) = (\hat{x}_i, \hat{y}_i, w_i, \sigma_i, x_i, y_i, x_j, y_j)$ where $(\hat{x}_i, \hat{y}_i)$ is the center of ribbon (vote center) and (x_i, y_i), (x_j, y_j) are edge pairs (vote sources). In Figure 5.4 in the first image, estimated pdf is shown with iso-contour lines. On the other figures, each color represent a different object. In Figure 5.4.a we plotted the edges (vote sources) with white color and vote centers with red markers. We also plotted the estimated pdf in iso-contour lines and local maxima with green triangle markers. As expected, the local maxima are at the vicinity of the center of each building. Due to our voting rule, non-rectangular shapes (such as road edges) have votes but with low probability. We also used other information such as corners, shadows, and steerable filter. Thus, the roads and other clutter objects have low probability. Therefore, they are not detected as buildings.

For each mode of the estimated pdf, we will only select the votes that are around the modes with an adaptive threshold. We could select the votes which are closer than a constant distance r to each local maximum. However, it would be problem for small and large buildings. Therefore, we use an adaptive distance that is extracted for each local maximum using the probability value at $p_B(\hat{x}_{B_k}, \hat{y}_{B_k}|F)$.

In Algorithm 5.1, the steps of the back-projection algorithm are listed. For each mode on $p_B(\hat{x}_{B_k}, \hat{y}_{B_k}|F)$, we find a region around $(\hat{x}_{B_k}, \hat{y}_{B_k})$ where the point locations within the region have a probability above a threshold $p_B(x, y|F) \geqslant p_{ref} \times \kappa$. Here κ is $\in (0, 1)$. We call this region as $A(x, y)$. In the given example, we plot $A(x, y)$ boundary with blue lines in Figure 5.4.b. Using the votes in that area, we back-project the votes to the vote sources (edge pairs) and find the shape for each building individually. Thus in the estimated pdf, only the votes contributed the most help to construct the shape of the buildings. In the example, we selected $\kappa = 0.75$. In Figure 5.4.b we give the back-projected votes where for each building it is given with different color. Thus, handling each local maximum of pdf individually, each building outlines are extracted successfully.

Now that we have the edges of each building and labeled them individually, we can extract a rough shape of the buildings. Here, we use a convex-hull algorithm [118]. The convex-hull of edge points will be the smallest convex outline that contains the edges.

We extract the convex-shape of the objects in two steps. First, we find the convex-hull of the edge points. Then, we convert the convex-hull polygon to a filled binary mask. We call this segmentation result as convex shapes. The result is given in Figure 5.4.c. Here, we paint each building with different color for better visualization. The convex shapes almost give the accurate shapes of the buildings. However, these are rough shapes and in some cases these shapes will be too rough if the shape of the object is not convex-hull such as L-shaped buildings. There will also be problems if we cannot extract the edges successfully.

Algorithm 5.1. Iterative shape extraction with vote back-projection

```
BEGIN
$K$; number of modes at estimated pdf (number of possible object centers)
$N$; number of votes
$L(x, y) = 0$; initialize the labeled image with zeros at all pixels
$\kappa$; set a local threshold value $\in (0, 1)$
$k = 1$; initialize the counter for buildings
Repeat for each mode $(\hat{x}_{B_k}, \hat{y}_{B_k})$
while $k \leqslant K$ do
  $p_{ref} = p_B(\hat{x}_{B_k}, \hat{y}_{B_k}|F)$; Extract a reference probability
  $A(x, y)$; Find a 2D region around $(\hat{x}_{B_k}, \hat{y}_{B_k})$ where the point locations
  within the region have a probability with $p_B(x, y|f_1) \geqslant p_{ref} \times \kappa$
  $i = 1$; initialize counter of votes
  while $i \leqslant N$ do
    if $(\hat{x}_i, \hat{y}_i) \in A(x, y)$; if $i^{th}$ vote center is in desired local area then
      $L(x_i, y_i) = k$; label the edge point
      $L(x_j, y_j) = k$; label the edge point
      $i+ = 1$; increase the counter for votes
    end if
  end while
  $k+ = 1$; increase the counter for buildings
end while
END
```

For example, at the right bottom corner of Figure 5.4.b we extracted some edge pixels that do not belong to the building. This caused a defect in convex shape result in Figure 5.4.c. To eliminate such errors and give better details of the shape, we use active contour segmentation method with the aid of initial convex shapes. Specifically, we use Chan-Vese active contours [119]. In this algorithm, when an initial mask (contour) is given and thus the mask evolves to a final shape and separates the image as background and foreground. The user can define if the tendency of the mask to shrink inwards or to grow outwards. We used our convex shape results as the initial mask of the active contour segmentation. One can utilize other segmentation algorithms also.

Here the purpose is using our convex shape masks, the segmentation results might get better. In Figure 5.4.d we give the Chan-Vese segmentation result for 30 iterations. As can be seen, the shape of the objects are better and the defect on some of the buildings shapes are corrected. When the iteration number is increased, one can obtain more detailed shapes.

The proposed algorithm here is not meant to be used for buildings only. Other simple objects shapes such as trees and ships are also extracted with the same method. In the experiments section, we give visual results for each object type.

5.1.2. Shape Extraction of Complex Objects

In the previous chapter, we defined airplanes and cars with the combinations of their different parts. Thus, their shape extraction requires the combination of the object's individual parts. In this section, we explain our shape extraction approach on airplane and car images.

5.1.2.1. Airplane Segmentation

We summarized the proposed method in Figure 5.5. There are two main differences from the simple shape extraction method. The first one is that we find the orientation of the airplane. For this purpose, for each local maximum location in pdf of airplane centers, we look for in which θ the pdf value of that local maximum is the largest in Eqn. 4.15. Then it is assumed as the airplane orientation.

After finding the airplane orientation, we back-project the votes to each part of the airplane. It is the second difference from shape extraction of simple objects. Then, for each part we extract the convex shapes and merge the results for full-shape extraction of an airplane.

We give each step of the algorithm on a sample image. In Figure 5.6.a we give a sample satellite image of airplanes. In Figure 5.6.b we give the Canny edges. Assume we estimated

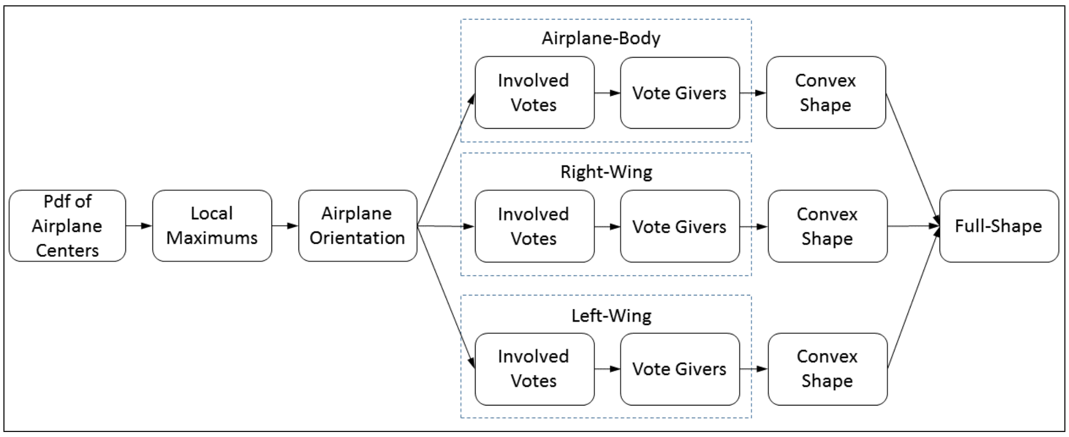


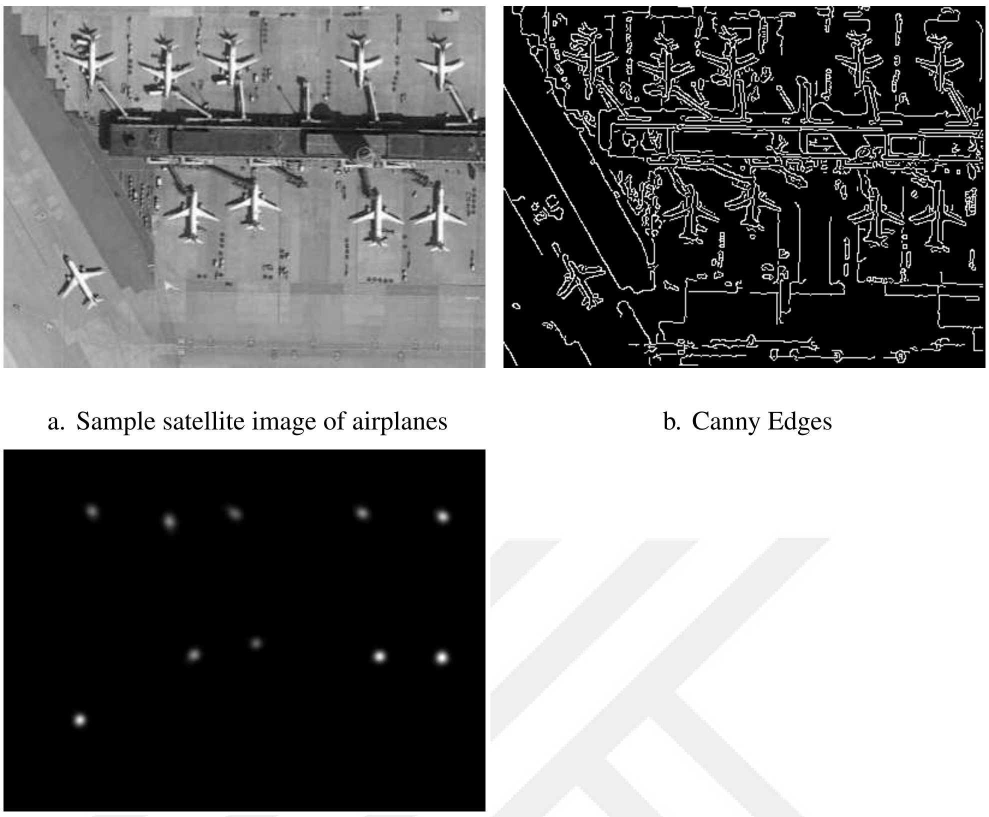
Figure 5.5. Steps of the vote back-projecting method for airplane shape extraction.

the pdf of airplane center locations with our method as explained in Section 4.3.1. The pdf estimation result is given in Figure 5.6.c. Here, we successfully detected all airplane centers.

As a reminder, the pdf estimation of airplane center involves the votes of three parts of the airplane as main body, right wing, and left wing. These three parts of the airplane are detected using edge based ribbon shapes. They voted according to a relative geometry of the airplane parts, without knowing which ribbon shape is which part of the airplane. However, since we know the location of votes and relative geometry of each of the airplane parts, we can back-project the votes from the airplane center to each airplane part center and then to the edges of the airplane parts. Then when we merge all these edges, we will roughly obtain the shape of the airplane.

In Figure 5.7.a, we give the back-projected edges for each individual parts of airplanes. In this figure, each extracted airplane part edges are shown with different color. Then at the second step, we extract the convex shape of each airplane part individually. Thus, we will have the rough shapes of the airplanes. The result is given in Figure 5.7.b.

Now that we have masked the airplane shapes, we can have more details on the shapes using the Chan-Vese segmentation method as in the previous section. The result is given in Figure 5.7.c. The details of the airplane shapes are better now. In the experiments section, we give examples of different cases of the proposed segmentation algorithm.

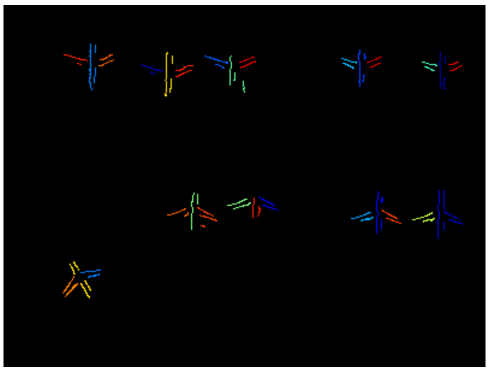


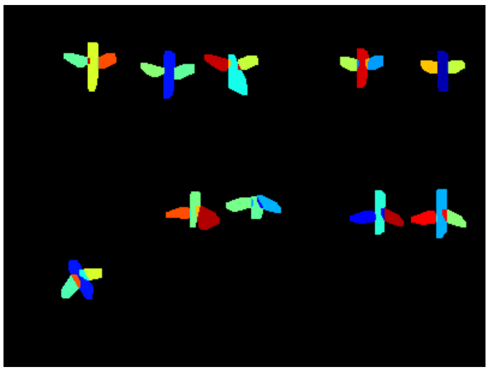
a. Sample satellite image of airplanes

b. Canny Edges

c. Estimated pdf of airplane centers

Figure 5.6. Sample satellite image of airplanes, Canny edges and pdf estimation.


a. Back-projected edges of each airplane part.


b. Convex shapes of back-projected edges.

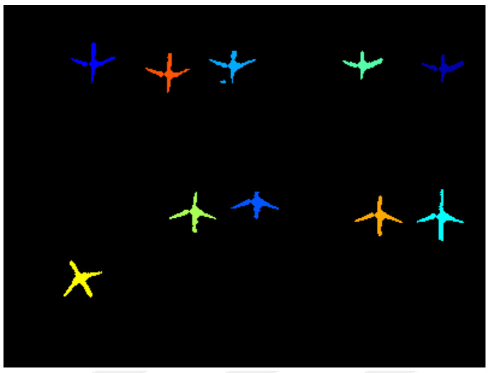

c. Chan-Vese segmentation when convex shape results are used as a mask.

Figure 5.7. Segmentation results for the image of airplanes.

5.1.2.2. Car Segmentation

In the previous chapter, we detected cars using relative geometry of its main body, windshield and its front part. We modeled it as combination of rectangles. In this section, we back-project the votes and extract the shape of cars similar to airplane shape extraction. We summarized the proposed method in Figure 5.8. The approach is the same with airplane shape extraction.

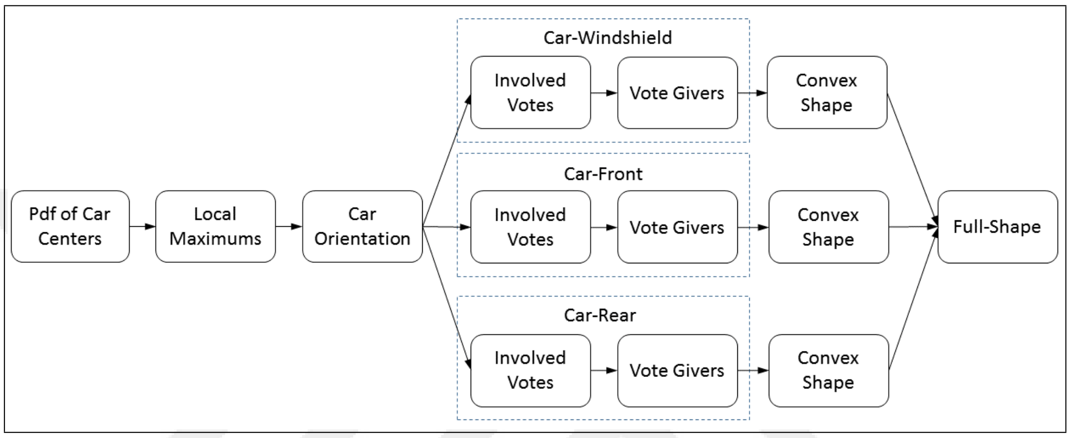
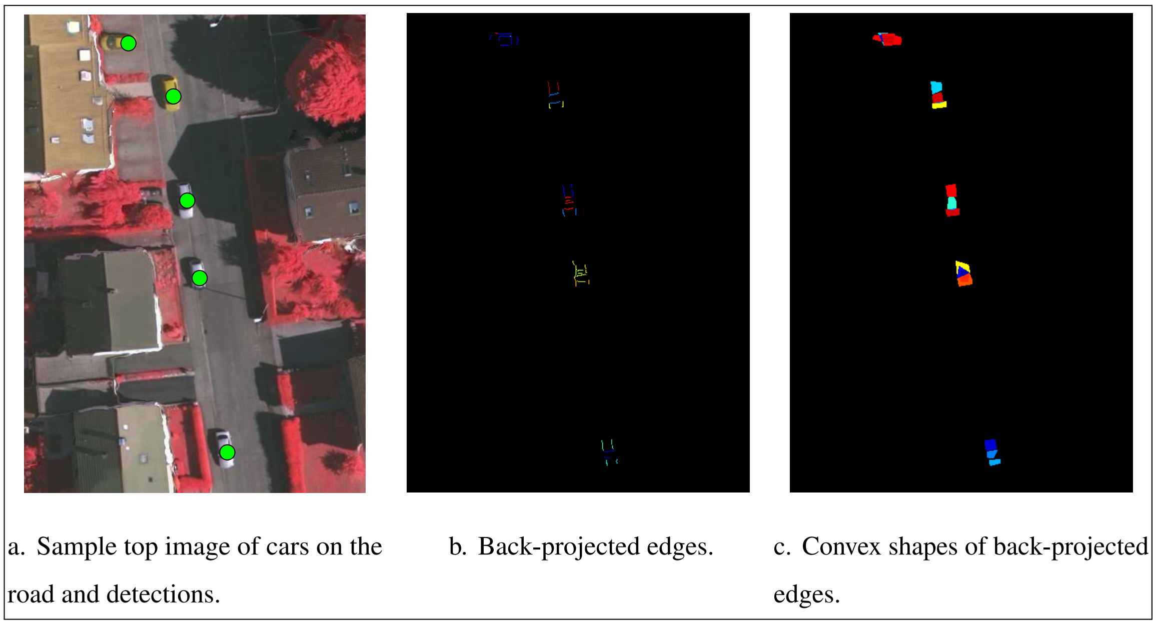


Figure 5.8. Steps of the vote back-projecting method for object outline extraction.



a. Sample top image of cars on the road and detections.

b. Back-projected edges.

c. Convex shapes of back-projected edges.

Figure 5.9. Segmentation results for cars.

We give an example on the same image of Figure 4.35.a. Here we detected five local maxima on the pdf of car centers. First, we find the orientation of each car by looking at each pdf of different θ values of Eqn. 4.29. Then, for each local maximum we back-project the votes to each part of the car and separately extract the vote sources. Extracted edges for each part is given in Figure 5.9.b with different color. Afterwards, we generate the convex shape of each part and merge them. The result is given in Figure 5.9.c where each part is given with different color. If the convex shape results intersect somehow, it is painted in a different color. As can be seen in Figure 5.9.c we detected three parts of each car. In the experiments section, we will give more visual results for different cases.

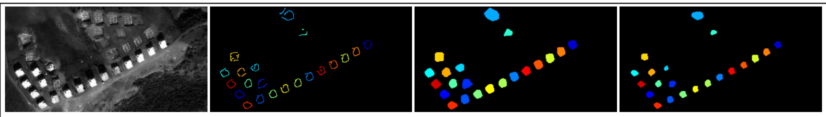
5.1.3. Experiments on Satellite Images

In this section, we give visual results of the proposed segmentation algorithm on various satellite images. We use the same data set as in Section 4.4. Since there are quite a number of image sets, we only give visual results for a subset of the test images.

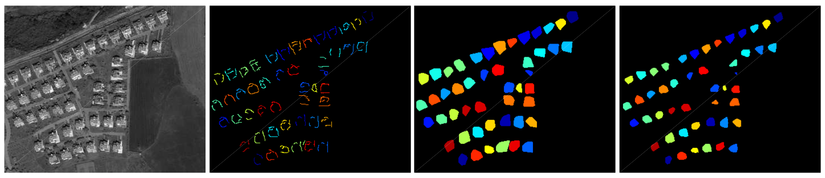
In Figure 5.10, we give visual results for the Ikonos satellite images of Adana. In this figure, there are six images. The first column represents the satellite images. The second column represents the back-projected edges. Here, the color of the edges denotes a separate object. In the third column, we give the convex shape of the back-projected edges. In the last column, we give the Chan-Vese segmentation result when the convex shape results are used as an initial mask. In the Chan-Vese segmentation method, we set the iteration number as 30 and tendency of the contour grow to inwards. We assume that our convex shape segmentation results are object pixels and other pixels as background. Hence, we assume the binary ones of the of Chan-Vese segmentation result as objects and binary zeros as background.

The Chan-Vese segmentation gives finer shape for buildings. However, in some cases the initial mask (convex shape) of a building shrinks too much and results in removing of the object pixels entirely. There are such examples in $Adana7$ and $Adana12$ images in Figure 5.10. Also in some cases, the segmented pixels are not removed completely but the outline of the object is shrunk. Since we labeled each of the object separately in the convex shape results, one can add a control to limit the area of Chan-Vese segmentation result at each iteration.

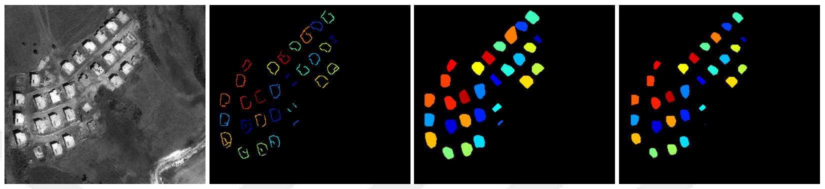
Next, we give visual results for the orchard tree image sets in Figure 5.11. In some cases, the



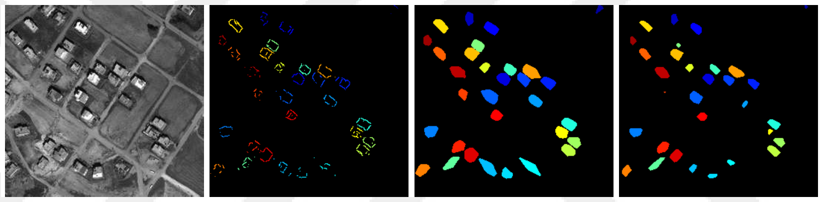
a. Adana1



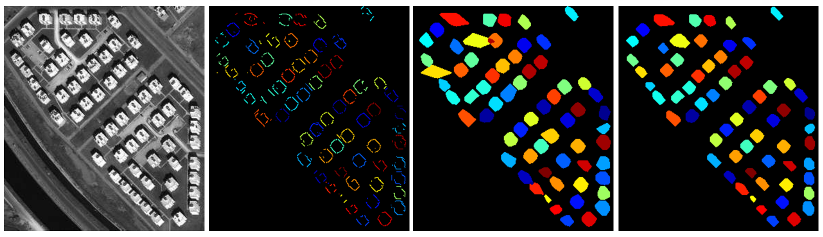
b. Adana6



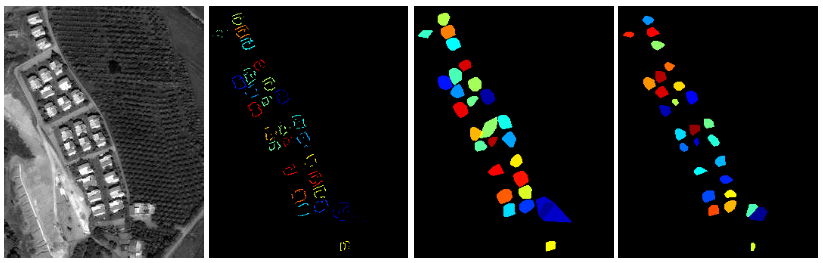
c. Adana7



d. Adana12



e. Adana13



f. Adana15

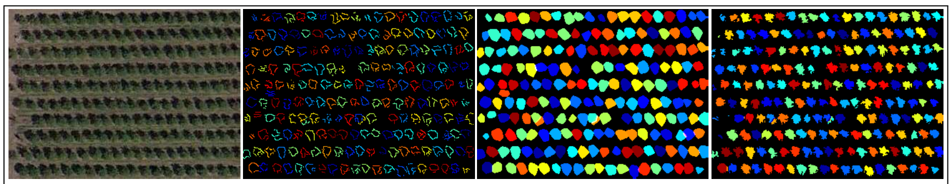
Figure 5.10. Segmentation results on some of the Ikonos satellite images of Adana.

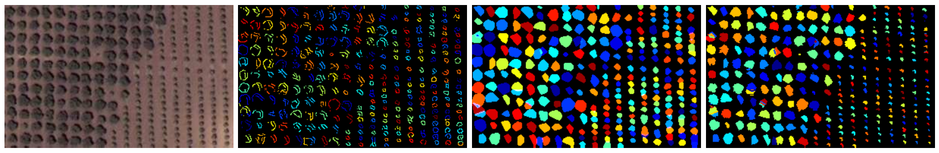
trees are located close to each other. Also, the color of the trees and the background varies in the same image. In Table 4.3 we give the spatial resolution and other information for each of the image. The tree diameters change from 2 to 20 pixels. We successfully separated the edges of the most of the trees individually. Even though the convex shape results are rough, Chan-Vese segmentation results give better detail on the tree crown sizes. The advantage of the proposed method is that it does not need multispectral data or shadow information in operation. Moreover, it does not depend on a specific tree type or background color.

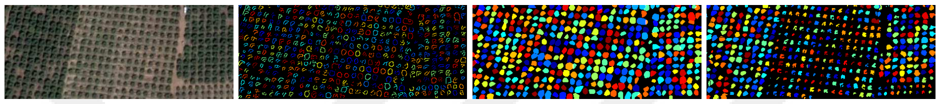
In Figure 5.12 we give the segmentation results on satellite images of ships. As a reminder, we used steerable filter features only for the estimation of ship center pdf. Thus we back-project steerable filter results to segment each ship pixels individually. In the second column of Figure 5.12, we give the back-projected votes. As can be seen, a single ship mostly has two lines at the left and right side of the ship. When we use these in our convex shape segmentation approach, the results are given in the third column. Here each color represents a different ship. As a reminder, in Figure 5.12.a there are 722 ships. Majority of the ships touch each other. Our method successfully detected 641 ships in this image and in which we also extracted the shapes of these ships. Here we do not use Chan-Vese segmentation algorithm. The length of the ship is only a few pixels in some cases. In most cases, the ships are in white color with some dark spots on the ship because of the shadow of the pole and sail. These make it difficult to successfully detect the ships pixels with the Chan-Vese segmentation method. Just after a few iterations, majority of the ship pixels are removed.

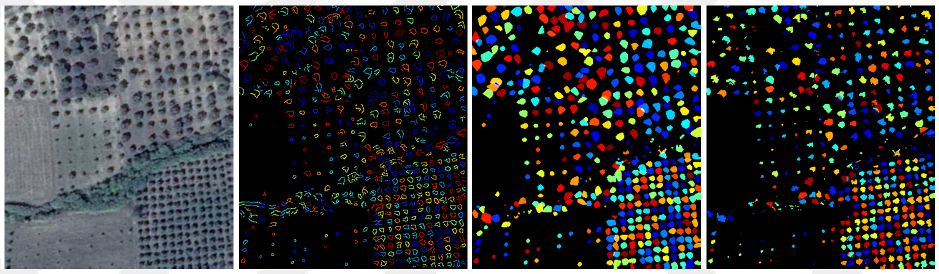
In Figure 5.13 we give the segmentation result of airplanes. In Table 4.8 we gave the metadata of the images. Here we show visual results for six of the images which are from six different airports. As in our airplane model in Figure 4.7, we successfully extracted the three parts of the airplanes in most cases. The results are given in the second and third columns of Figure 5.13. In the second and third columns, we paint each part of the airplane with different color. After using the Chan-Vese segmentation, we get better detail of the airplane shapes where we give the results in the last column of Figure 5.13.

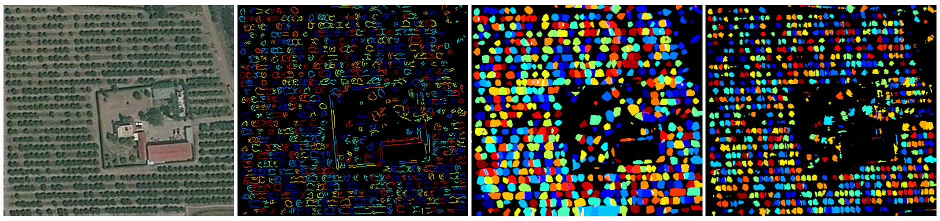
The majority of the false detections are caused by the jetway (jet bridge) which is a movable connector to extend the airport terminal gate to the airplane. When the jetway is connected


a. Image 1


b. Image 4


c. Image 5


d. Image 6


e. Image 9

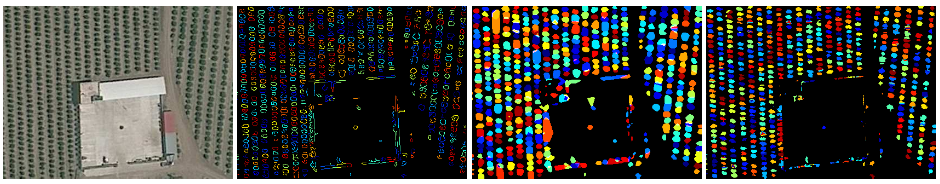
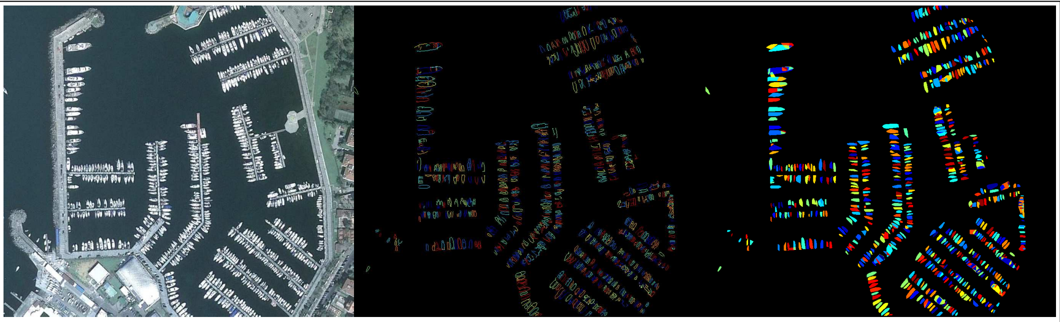
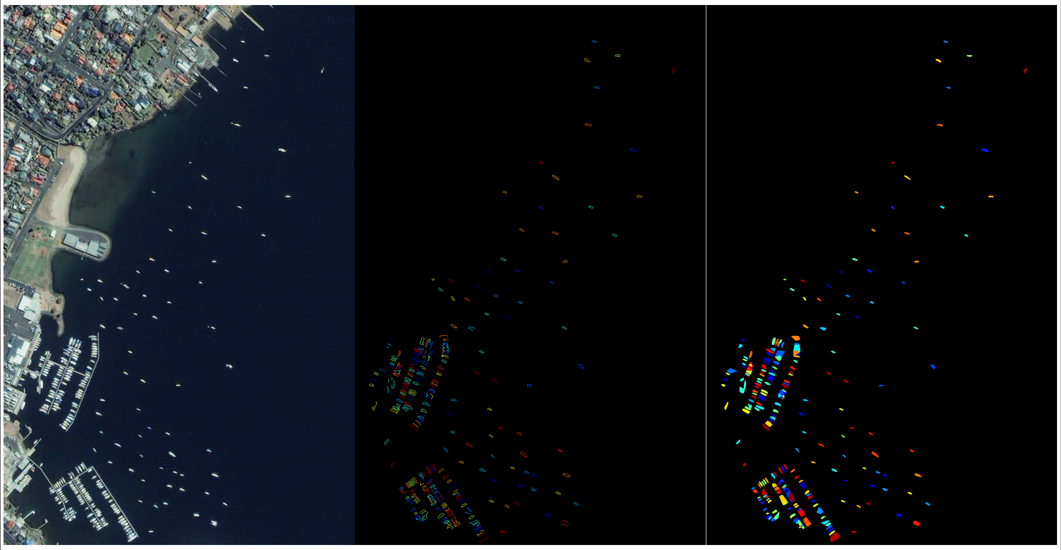

f. Image 10

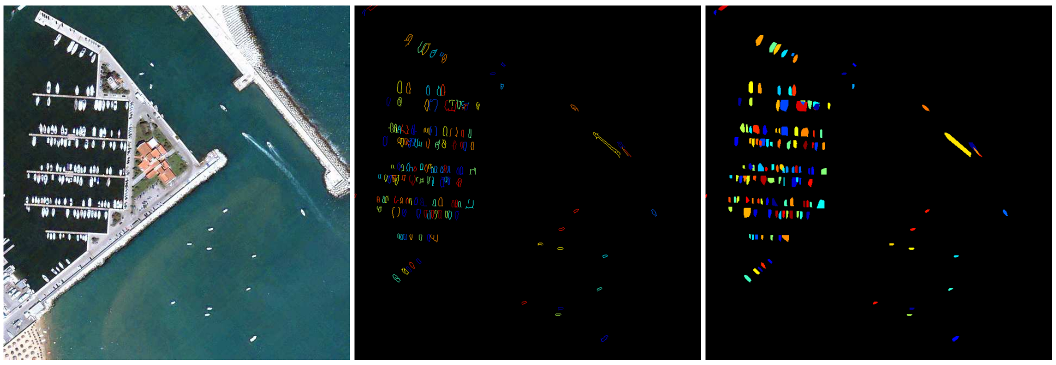
Figure 5.11. Segmentation results on some of the satellite images of orchard trees.



a. Geoeye-2

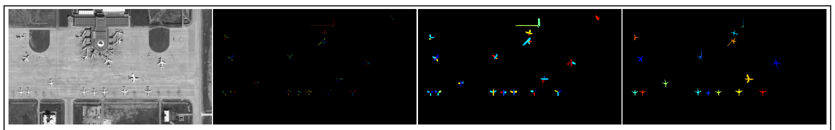


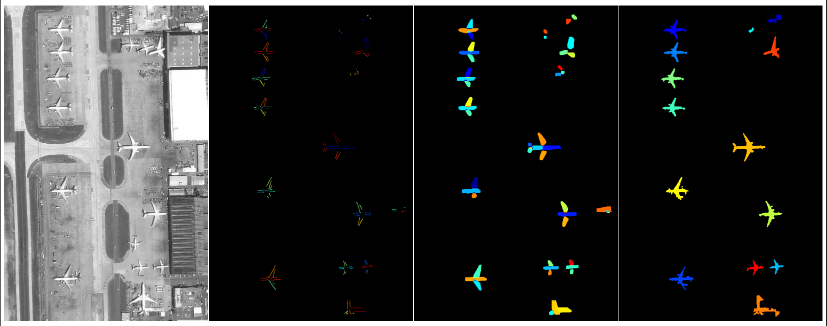
b. Ikonos-1

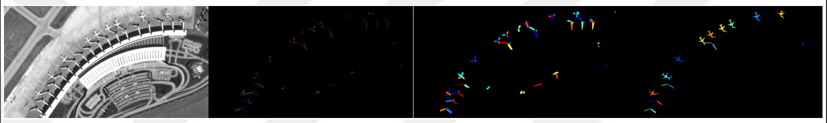


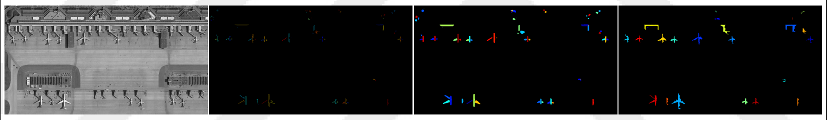
c. Quickbird-1

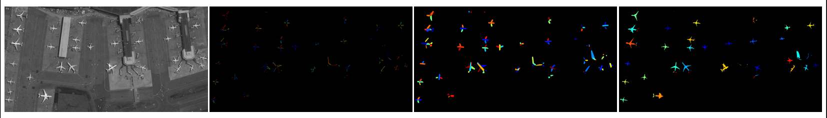
Figure 5.12. Segmentation results on some of the satellite images of ships.


a. Image 2


b. Image 4


c. Image 6


d. Image 8


e. Image 10

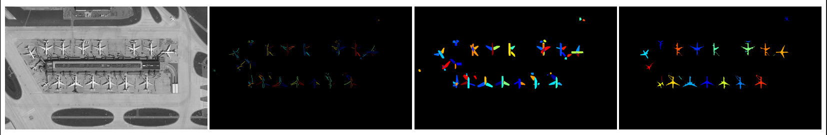

f. Image 12

Figure 5.13. Segmentation results on some of the satellite images of airports.

with the airplane, there we detect it as another airplane or in the segmentation process, it is detected as a part of the airplane. We see such results mostly in Image 6 in Figure 5.13.

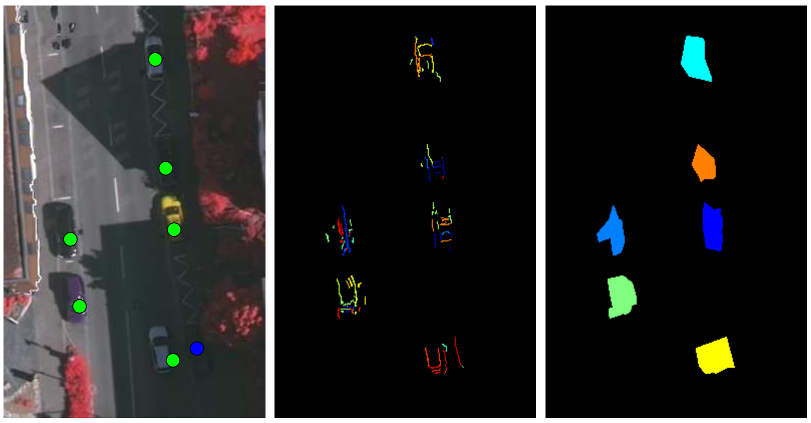
We tested the car segmentation method on ISPRS Vaihingen data set, which is explained in more detail in Section 4.3.2. Since the image patches are quite large in size, we only give the visual results for the subset of the images in Figure 5.14. In the first column we give the TOP images with TP detections in green, FP detections in red and FN detections in blue markers. In the second column, we give the back-projected edges of each part of the car. Each part is shown with different color. In the last column, we give the full convex shapes of cars.

In some cases, we could not detect some of the cars. For example in Figure 5.14.a a black car is in the shadow next to a white car. In this case, we could not detect the edges of the black car sufficiently because of its color and shadow. However, we could detect the outer edges of the car. In the voting process, the edges of the white car wrongly used the black cars outer edges. Thus the local maximum is shifted towards to the black car and we do not see any detection on the black car. It also affects the back-projecting step. As can be seen in Figure 5.14.a one of the extracted lines belong to the black car. So the extracted outline of white car's width is almost same with its length.

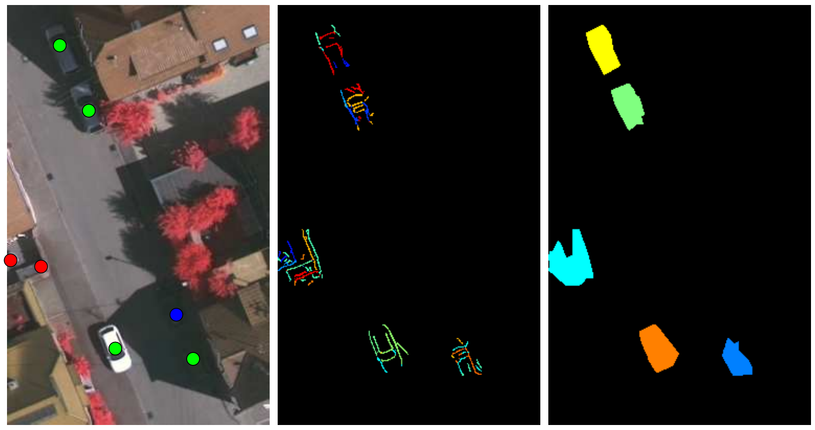
In some cases, we see some FP detections. These detections mostly caused by similar rectangular shapes of other objects such as a short wall on a sidewalk. Such two cases seen on the left side of Figure 5.14.b and about the center of Figure 5.14.d. In these cases, we used votes in the height model because the wall's elevation is similar to the cars. We also see shadows behind the walls which were wrongly assumed as a windshield of a car. Thus these spots accumulated votes and detected as a car. The extracted shapes of these objects also look similar to a cars shape.

5.2. SEGMENTATION OF HEIGHT DATA WITH VOTING AND MORPHOLOGY

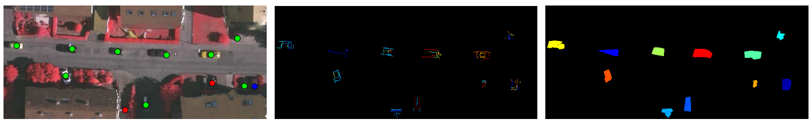
In this section, we segment height data using morphological operations. Here, we use height data only. LiDAR sensors provide height data from Earth's surface. This is a valuable source to solve several remote sensing and geospatial analysis problems [120–123]. Unfortunately,



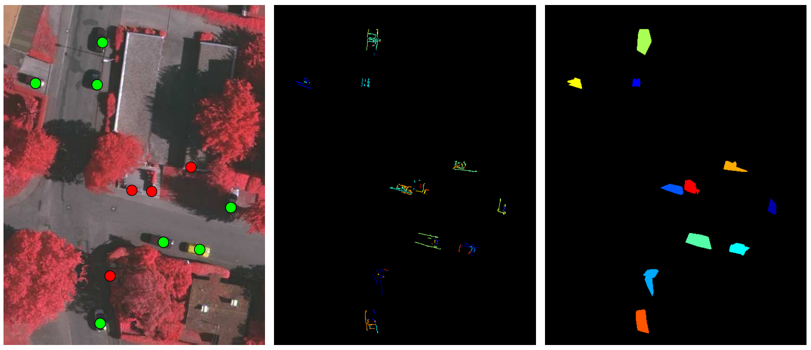
a. $top\_mosaic\_09cm\_area1$, $Subset_1$



b. $top\_mosaic\_09cm\_area1$, $Subset_2$



c. $top\_mosaic\_09cm\_area3$, $Subset_1$



d. $top\_mosaic\_09cm\_area3$, $Subset_2$

Figure 5.14. Segmentation results of cars on ISPRS Vaihingen data set.

LiDAR data includes ground and above-ground object returns together. To use it efficiently, it should be filtered as ground and non-ground points. Digital Terrain Model (DTM) generation is possible if only true ground points are interpolated. Non-ground points will lead to above ground object (such as buildings, trees, and cars) detection. These can be used to solve remote sensing and geospatial analysis problems in an effective manner.

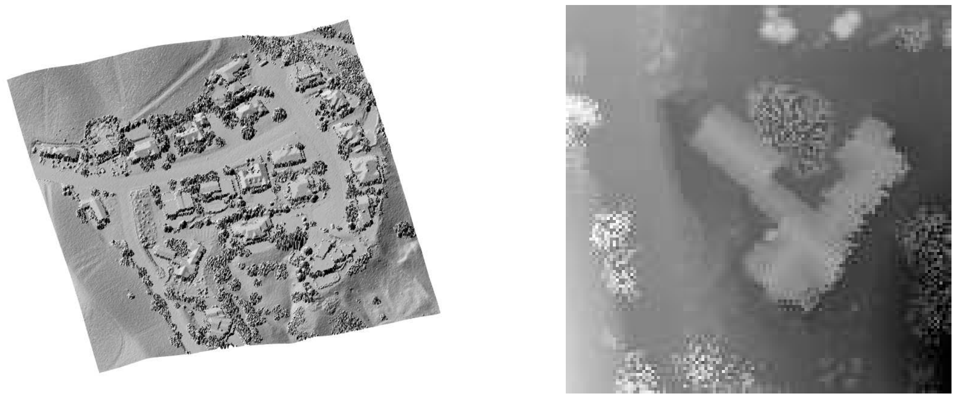
New LiDAR sensors provide higher spatial resolution data with wider coverage area. Hence, it is becoming more difficult to manually process the obtained data. Therefore, researchers proposed several methods to automate LiDAR data processing. The first step in the proposed method is detecting the most probable object center points. However, before that we need a preprocessing step to clean the data and obtain small objects initially.

5.2.1. Preprocessing

The proposed method works on gridded raster data. Therefore, if the LiDAR point cloud is available, it should be represented in gridded form. Here the nearest neighbor interpolation is used to generate the raster DSM from the LiDAR point cloud.

We will explain the proposed method on the Utah-5 DSM test data throughout the paper. This test sample (with spatial resolution of 0.5 m) is obtained from the NSF Open Topography website [124]. Here only the first pulse return is available. We provide the original Utah-5 test data without preprocessing in Figure 5.15.a. As can be seen there, it contains buildings and various vegetation on a sloped terrain. More detail on this test data can be found in the experiments section. We also picked a subpart of the Utah-5 test data as in Figure 5.15.b to explain some steps of the proposed method better.

Unfortunately, raw LiDAR measurements are often noisy due to sensor characteristics and other measurement errors. Specifically, most first and some last LiDAR pulse returns may cause irregular measurements on trees since the sent LiDAR pulse is reflected between tree branches. Sometimes, parts of the sent pulse don't gets back to the sensor. Therefore, the tree representation by first pulse data is not fully correct. For these reasons, morphological opening by a 5×5 disk is applied on DSM as $I_s = I \circ B$ where I is the DSM, $\circ$ is the



a. DSM data b. Subpart of DSM

Figure 5.15. The original Utah-5 DSM test data without preprocessing.

morphological opening operation and B is the 5×5 disk shaped structuring element. This operation removes some of the small objects and outliers in DSM data. Removed objects are elevation thresholded by

$$O_s = (I - I_s) \geq t_h \tag{5.1}$$

where O_s is the binary class of small objects and t_h is the elevation threshold. Therefore, some trees and small objects (such as cars and low vegetation) are detected in advance. Stereo image based DSM data also contains outliers and small objects. Therefore, if stereo DSM will be used, the same morphological opening operation should also be applied to it.

We first apply the preprocessing step to the Utah-5 test data and provide the obtained results in Figure 5.16.a. We also provide the small object detection results from this image in Figure 5.16.b. We will benefit from these in the following sections.

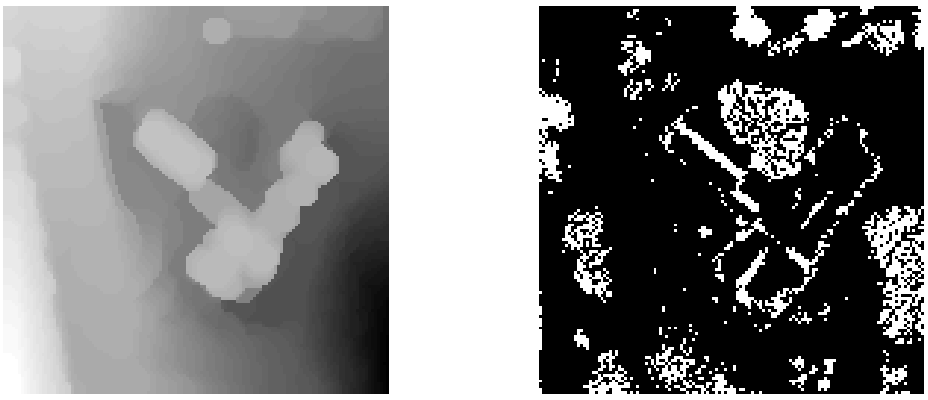


Figure 5.16. Demonstration of the preprocessing and small object detection steps.

5.2.2. Initial Segmentation

In Chapter 5, the height data is used in the probabilistic object center location voting framework. We used edges of DSM in the voting process and obtained the object center probability map. The first step in DSM segmentation is based on the modes of this pdf. These will be used in extracting reference elevation values. Let's say M modes are extracted. These can be represented as (x_{B_i}, y_{B_i}) for $i = 1, \cdots, M$. An initial segment can be obtained with connected pixels to each (x_{B_i}, y_{B_i}) in an iterative manner as

$$X_k = (X_{k-1} \oplus B) \cap S \tag{5.2}$$

where,

$$S = \sum_{i=1}^{M} |I_s(x_{B_i}, y_{B_i}) - I_s(x, y)| < t_h \tag{5.3}$$

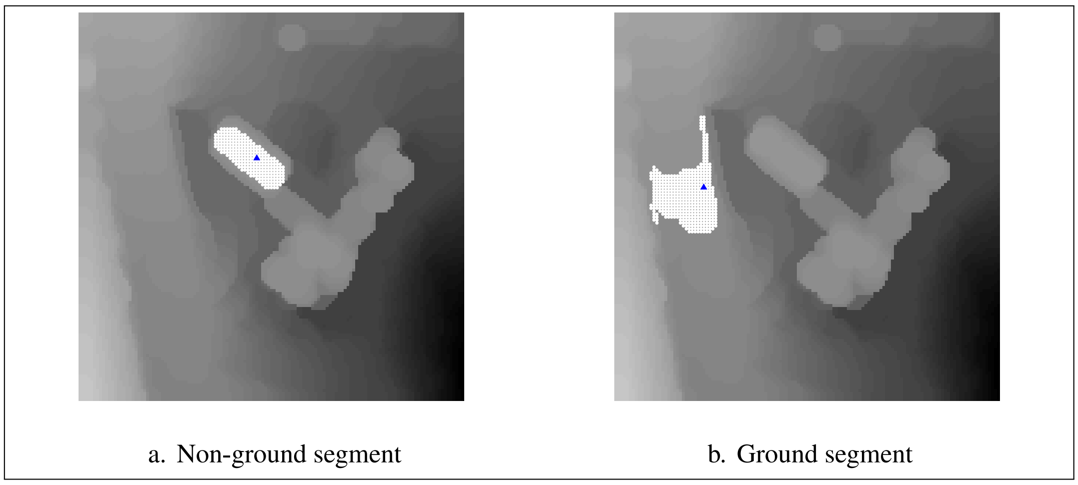


Figure 5.17. Initial segmentation results on non-ground and ground.

where $X_0 = S(x_{B_i}, y_{B_i})$, $\oplus$ denotes the dilation operation, and B is a 3×3 structuring element [125]. If $X_k = X_{k-1}$, then the iteration stops.

Initial segmentation result on the subpart of Utah-5 test data is given in Figure 5.17. Here white dots indicate initial segmentation results. Two cases are given here. In the first case, the mode is on a building. The initial segmentation result for this case is given in Figure 5.17.a as white dots. As can be seen, the initial segment partly overlaps the building. In the second case, the mode is on a sloped region. Initial segmentation result for this case is given in Figure 5.17.b as white dots. As can be seen there, the initial segment spreads on the ground with the same elevation value.

Algorithm 5.2. Morphological region growing on DSM data

```
BEGIN
$(x_i, y_i) \in X_k$; points in the initial segmentation
$\mu_X = \frac{1}{N}\sum_{i=1}^{N} I_s(x_i, y_i)$; the mean elevation value
$t_l$, $t_u$, $t_s$; lower, upper, and final elevation threshold values
$T_{max}$; maximum iteration number
$B_s$; sequence of 3 $\times$ 3 structuring elements for thickening
$\circledast$; hit or miss transform
$\ominus$; erosion operation
$counter = 0$; initialize the counter
repeat
  $X_k^+ = X_k \cup (X_k \circledast B_s)$; thicken the current segment
  $X_b^+ = X_k^+ - (X_k^+ \ominus B)$; extract the boundary of the thickened segment
  $\Delta\mu_k^+ = \mu_X - I_s(x_b, y_b)$ where $(x_b, y_b) \in X_b^+$; the elevation difference for each
  boundary pixel
  if $\Delta\mu_k^+ > t_u$; if any boundary pixel's elevation value is too low then
    $X_k^+(x_b, y_b) = 0$; set the related pixel to zero
  end if
  if $\Delta\mu_k^+ < t_l$; if any boundary pixel's elevation value is too high then
    $X_{k^+}(x_b, y_b) = 0$; set the related pixel to zero
  end if
  if $X_k^+ = X_k$ or $counter > T_{max}$ then
    $EXIT = true$; stop growing
  else
    $EXIT = false$; keep growing
    $X_k = X_k^+$; update the initial segment
    $\mu_X = \frac{1}{N}\sum_{i=1}^{N} I_s(x_i, y_i)$; update the mean elevation value
    $counter+ = 1$; increase the counter
  end if
until $EXIT$
$\mu_b = \frac{1}{M}\sum_{i=1}^{M} I_s(x_{b_i}, y_{b_i})$ where $(x_b, y_b) \in X_b^+$; outer perimeter mean elevation
$\Delta\mu = \mu_X - \mu_b$; elevation difference of the object and its outer perimeter
if $\Delta\mu < t_s$; if not satisfying being an object then
  $X_k(x_i, y_i) = 0$; delete all pixels
end if
return $X_k$
END
```

5.2.3. Morphological Region Growing

The initial segmentation result will be used in the morphological region growing method to extract possible non-ground object points. The proposed method is algorithmically explained in Algorithm 5.2. This method can be summarized as follows. The outer boundary of the initial segment is added to the segmentation result at each iteration. This is done by checking the elevation value of each pixel in the outer boundary. The final segmentation result is obtained if the mean elevation height of the segment is greater than the mean elevation height of the segment's border pixels. This method is performed for all modes of the vote map. Finally, all segmentation results are merged.

The iteration steps of the proposed morphological region growing algorithm on the initial non-ground segment (given in Figure 5.17.a) is given in Figure 5.18. In this figure, white dots indicate the segmentation result at each iteration. Red dots are pixels whose height is lower or higher than the upper and lower thresholds respectively. As the region grows at each iteration, the mean height of the segment is updated. Hence, if the non-ground object is a building and has a sloped or flat rooftop, the algorithm can handle it. As the iterations end, the algorithm checks whether the segmentation result belongs to a non-ground object or not. For the case in Figure 5.17.a, it belongs to a non-ground object. Therefore, the final segment is accepted. Iteration steps of the proposed region growing algorithm on the initial ground segment (given in Figure 5.17.b) is given in Figure 5.19. For this case, the final segment is rejected.

5.2.4. Final Non-ground Object Segmentation

In order to detect all objects and extract their shape in DSM data, the algorithm in Algorithm 5.2 is applied on all modes of pdf. As corresponding segments are obtained, they are merged with the initial small non-ground object detection result by a logical "or" operation [126]. This step can be formulated as

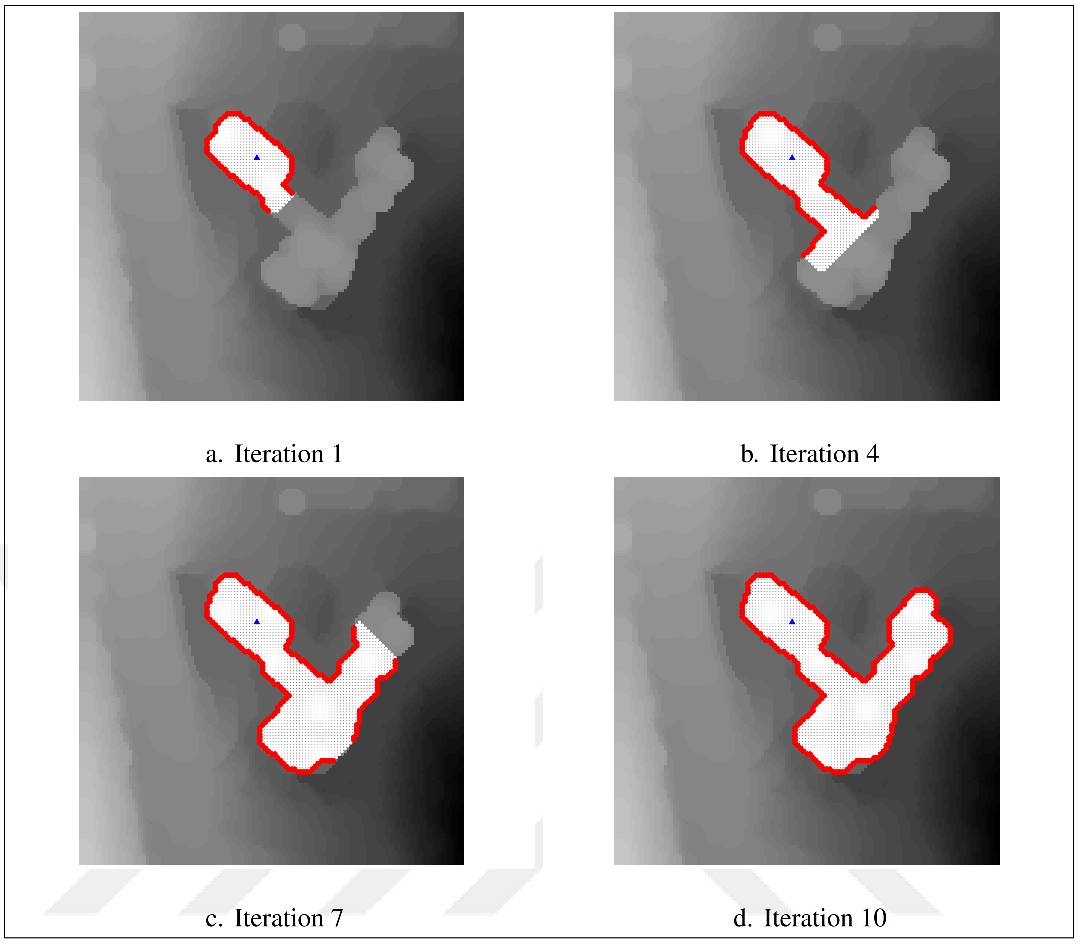
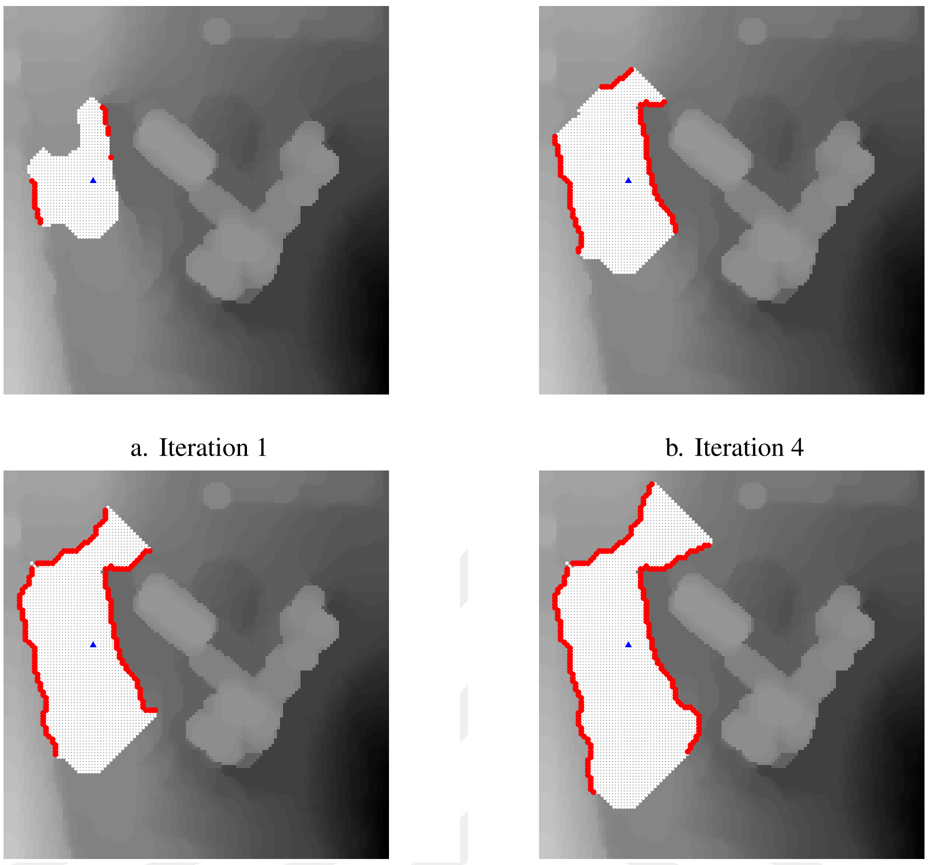


Figure 5.18. Mmorphological region growing algorithm on the initial building segment.

$$O = O_s \cup \sum_{j=1}^{M} X_k(j) \tag{5.4}$$

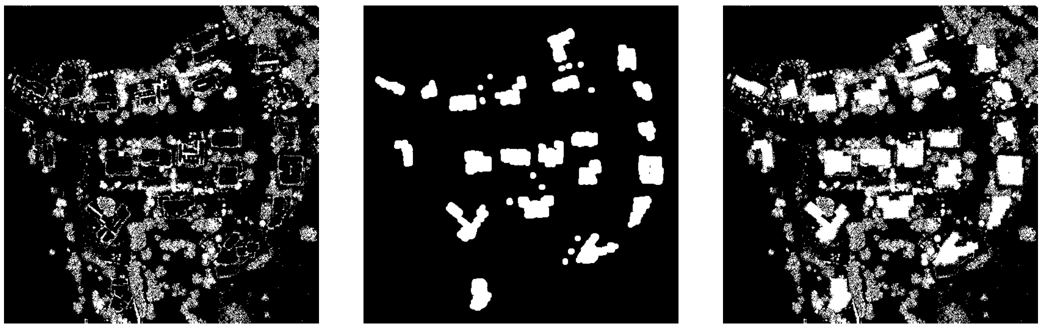
where O is the final object detection result. O_s is the small non-ground object detection result obtained in the preprocessing step. $X_k(j)$ is the segmentation result for the j^{th} mode of pdf.

Object detection results for the Utah-5 test data are given in Figure 5.20. Small object detection result is given in Figure 5.20.a. Segmentation via morphological region growing is given in Figure 5.20.b. Their combination is given in Figure 5.20.c. As can be seen in this figure, most of the objects are detected in this test data. Quantitative results for the Utah-5



a. Iteration 1
b. Iteration 4
c. Iteration 7
d. Iteration 10

Figure 5.19. Morphological region growing method on a ground segment.



a. Small non-ground objects
b. Morphological region growing result
c. Final non-ground objects

Figure 5.20. Non-ground object detection results for the Utah-5 DSM test data.

and other test data will be given in Section 5.2.6.

5.2.5. DTM Generation

In previous steps, non-ground points are extracted by segmentation. At this step, the remaining points are taken as ground points. Hence, DTM can be generated from these. Here, we use image inpainting to generate DTM as suggested by Pingel *et al.* [100] and Özcan *et al.* [127]. In image inpainting, missing pixels of the image are interpolated with existing neighbor pixels [128]. Similarly we label the non-ground pixels in DSM as missing and fill them with Derrico's [129] image inpainting method in order to obtain the DTM. The true DTM and generated DTM for Utah-5 is given in Figure 5.21. As can be seen in this figure, the generated and true DTMs are similar.

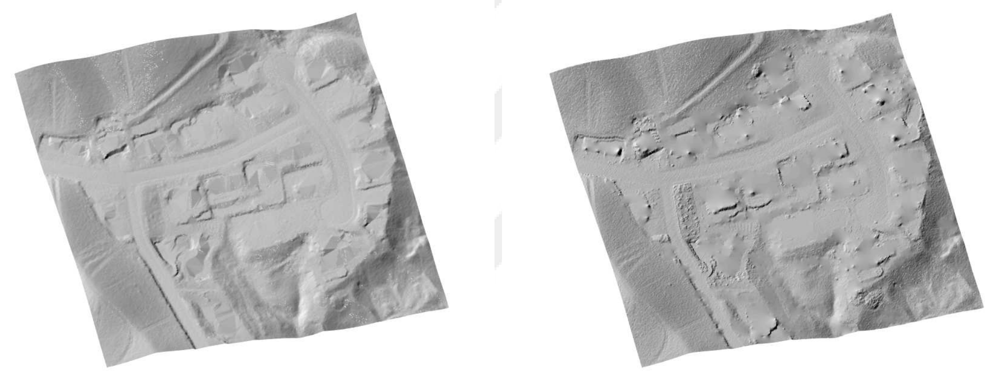

a. True DTM b. Generated DTM

Figure 5.21. DTM for the Utah-5 DSM test data.

5.2.6. Experiments on 3D Data

The proposed method is tested on three different data sets. The first data set is obtained from the publicly available NSF Open Topography website which provides LiDAR based DSM data [124]. Challenging residential regions are selected from the web site and the test set is formed accordingly. The proposed method is also compared with three other methods on this data set. The second data set is composed of DSM data generated from stereo WorldView image pairs. Industrial and residential regions are selected here. The third data set is obtained from ISPRS which is used as a benchmark [83].

For all data sets, the performance of the proposed method is provided in terms of non-ground object detection results. We use several evaluation metrics are used. Specifically, Kappa (K) score, total error (TE), type-I error (TI), and type-II error (TII) are used for the first and second data sets [83, 130]. Completeness (CP), correctness (CR), and F1 score metrics are used in the third data set to be consistent with other methods using this data [131].

5.2.6.1. Parameter Settings

There are three data sets as mentioned previously. Spatial resolution is 0.5 m for the first two data sets. Spatial resolution is 0.09 m for the third data set. Next, parameter values for the first two data sets are provided. These values should be rescaled according to the resolution of the third data set. Through the voting and segmentation steps it is assumed that a non-ground object is at least one meter above the ground. Therefore, the elevation threshold in Eqn. 5.1 is selected as $t_h = 1$ m. This value may be increased or decreased with respect to the height of small objects to be detected. When a non-ground object is larger compared to the window size in Eqn. 3.23, votes may not accumulate at one location on the object; but may result in more than one accumulation point. This may cause multiple peaks in the estimated pdf. On the other hand, if the window size is selected too large, then some objects may not be detected. As a compromise, the voting window size in Eqn. 3.23 is selected as $w = 3$ m. The width of the Gaussian kernel in Eqn. 4.1 also directly affects the number of peaks in the estimated pdf as with the voting window size. Therefore, it is also set as $\sigma = 3$. To note here, if multiple peaks are formed in the estimated pdf, this will only increase the processing time since the proposed method starts segmentation at each mode point. Besides that, there will be no difference in the obtained result.

The iteration number in Algorithm 5.2 is set as $T_{max} = 10$. Increasing T_{max} leads to longer processing time and does not significantly change the object detection result. Elevation threshold values in Algorithm 5.2 are set as $t_l = -5$ m, $t_u = 1$ m, and $t_s = 1$ m. The t_l value is set such that the segmentation step can continue with a higher object part. Here, the negative value only indicates the difference between the height values in the present and nearby object parts. Let's assume that an object is partly surrounded by a taller object where the height difference is lower than five meters. If region growing starts on the lower object, it

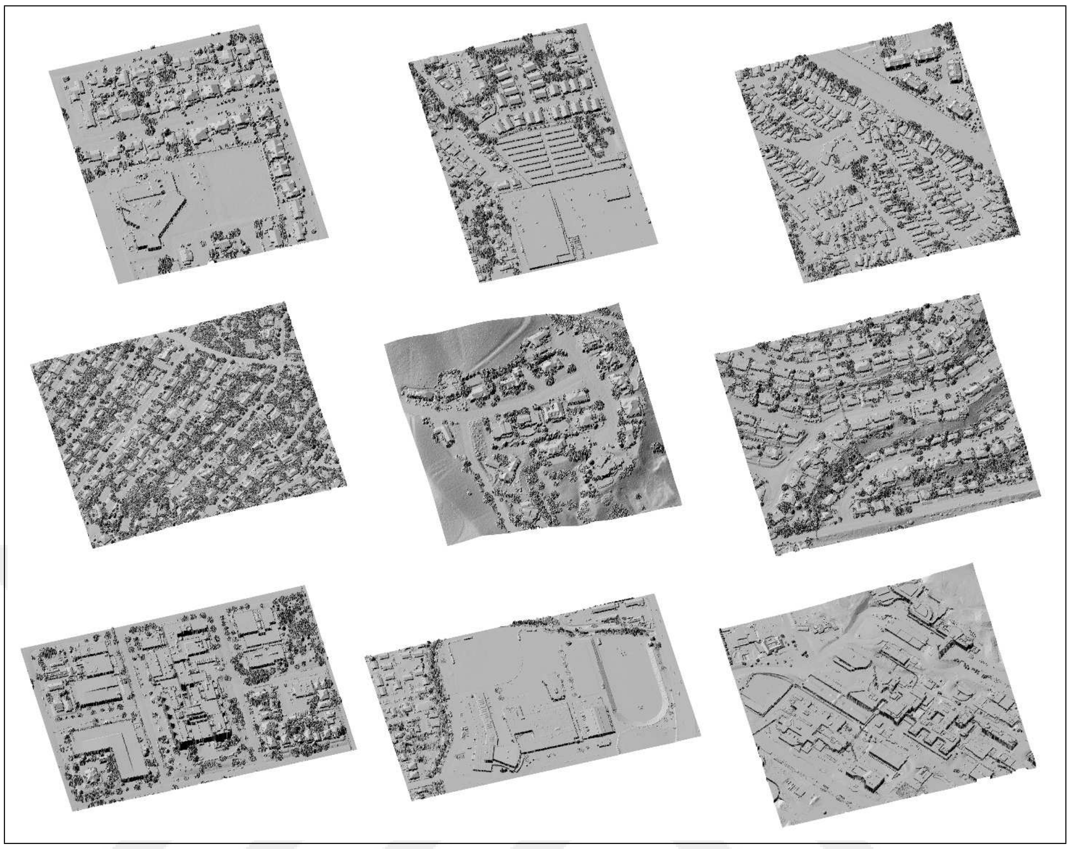


Figure 5.22. Utah DSMs. 1 row: Utah 1-3; second row: Utah 4-6; third row: Utah 7-9.

will continue growing on the taller object. If $t_l < -5$ m, then segmentation stops on the lower object.

5.2.6.2. NSF Open Topography LiDAR Data Set

The first data set is obtained from the publicly available NSF Open Topography website [124]. The reader can download LiDAR point cloud data by defining an area of interest from this website. The reader can also download DTM of the selected area from the mentioned website. Nine test samples are picked from the NSF Open Topography website. These are labeled as Utah 1-9 throughout the study. DSM data for all test samples are given in Figure 5.22.

The grid resolution for all test samples is 0.5 m. The DTM for this data set is also available on the website. Due to lack of manually generated ground truth for test samples, the downloaded DTM is used to form the ground truth. Therefore, the provided results in this section will be

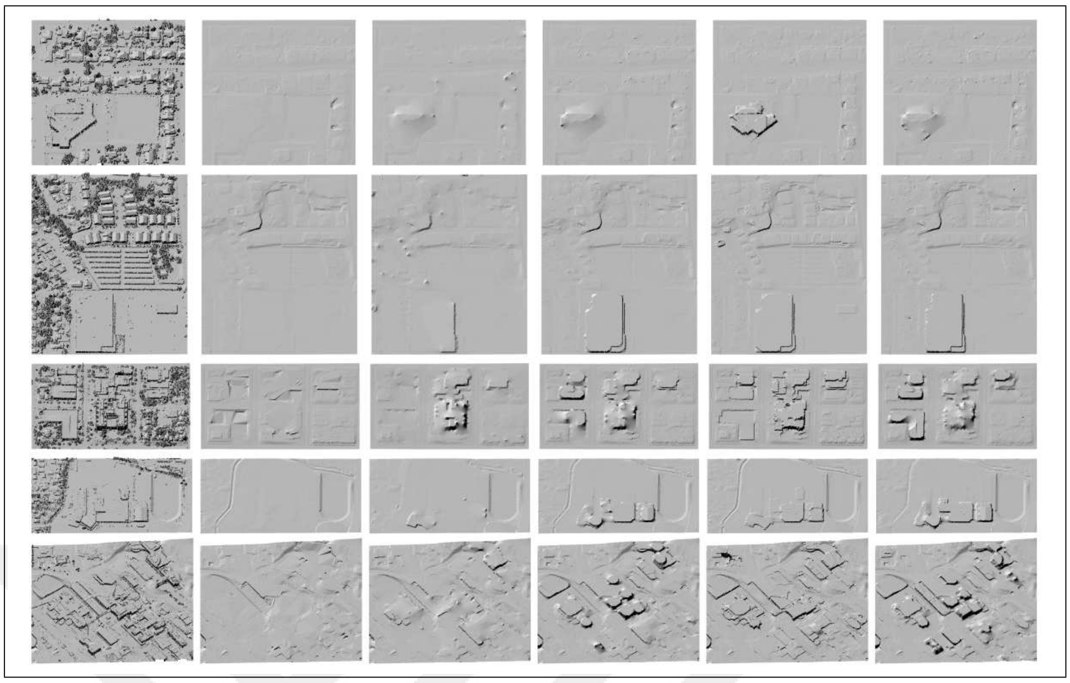


Figure 5.23. Generated DTMs for Utah-1, Utah-2, Utah-7, Utah-8 and Utah-9 test data.

relative to the downloaded DTM.

Test samples in this data set are acquired from residential regions. These are summarized in three cases according to the terrain type, object size and type as follows. Test samples Utah 1-3 have flat terrain; buildings here are mostly in one piece. Building layouts in these test samples change from small to large. There are trees on the side of roads and buildings. In Utah-1, there is one large building with an irregular shape. In Utah-2, there is one large building with flat roof. In Utah-3, buildings mostly have equal size with four of them having different size. On these samples, the proposed method is tested for detecting objects with varying size located in a flat terrain. Utah 4-6 test data have sloped terrain with regular sized separate buildings. In Utah-4, buildings are surrounded by trees. In Utah-5 and Utah-6, buildings have irregular shapes with trees nearby. On these samples, the proposed method is tested for detecting objects on a sloped terrain. In Utah 7-9, buildings are connected with varying size and some buildings exhibit irregular shape. In Utah-7 and Utah-8, the terrain is mostly flat. In Utah-9, the terrain is partly sloped. On these samples, the proposed method is tested for detecting large and connected objects on a flat or sloped terrain.

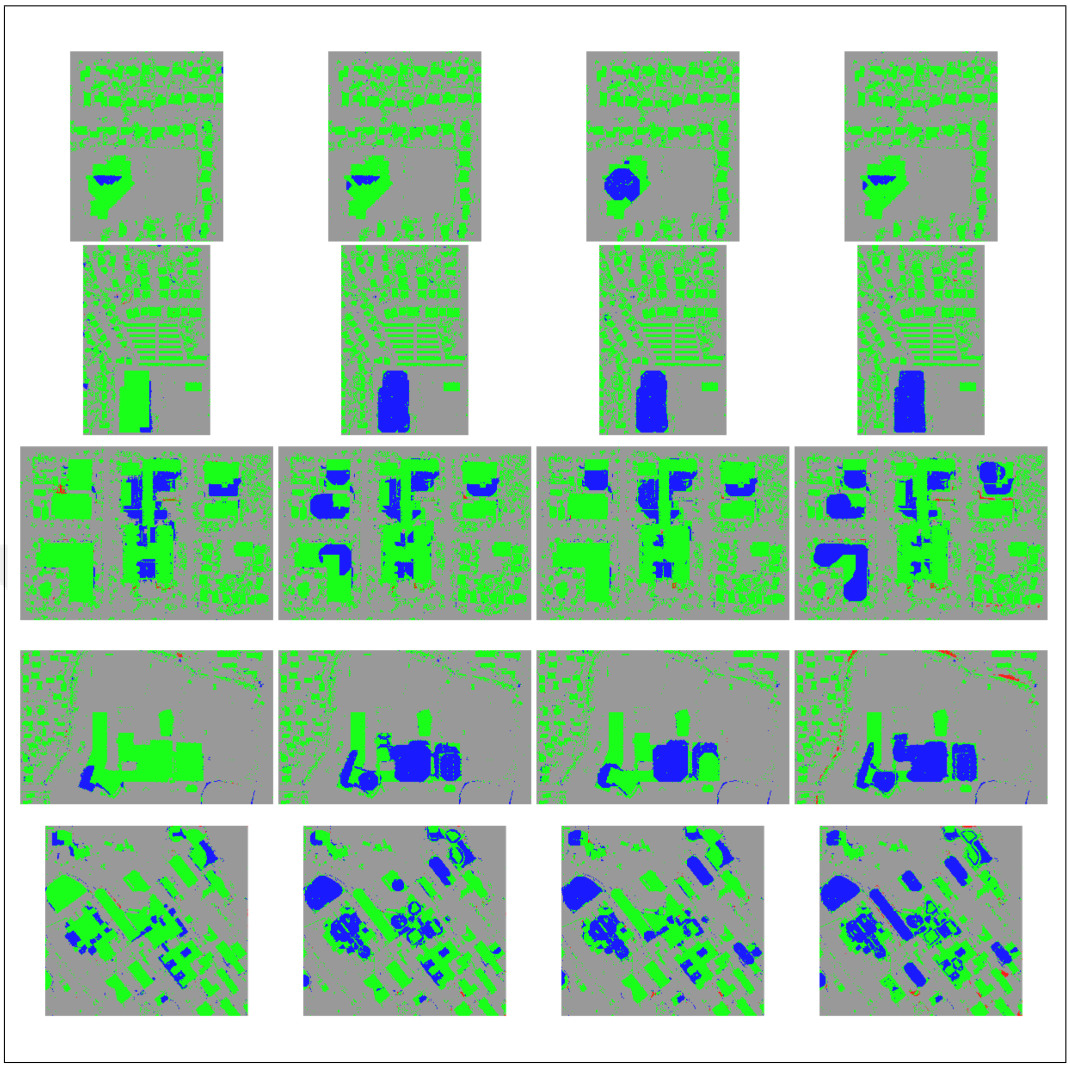


Figure 5.24. Detailed object detection results for the Utah 1-2-7-8-9 test data.

The proposed method is compared with three other methods in the literature using the NSF data set. The first method is introduced by Pingel *et al.* [100]. This method is based on iterative morphological filtering where it generates DTM and then detects objects on the normalized DSM. The software for this method is called simple morphological filter (SMRF) which can be obtained from [132]. Within this software, default parameter values are used as suggested in [100]. The second method is proposed by Mongus *et al.* [101]. This method is also based on morphological operations. It uses a top-hat scale-space transform using differential morphological profiles on point's residuals from the approximated surface. Surface and regional features are used for building detection. The software for this method is called gLiDAR and can be obtained from [133]. Within this software, default parameter values are used as suggested in [101]. The third ground filtering method is based on an iterative Empirical Mode Decomposition (EMD) [127]. For the assessment of these methods, ground truth object locations are formed as follows. First, the ground truth DTM is subtracted from DSM and the normalized DSM is obtained. Then, pixels above one meter are considered as belonging to a non-ground object. Remaining pixels are taken as ground. Since SMRF, gLiDAR, and EMD filtering methods also generate DTM, the same procedure is applied to detect non-ground objects.

Generated and true DTMs for Utah 1-2-7-8-9 are given in Figure 5.23 respectively. Here the first column corresponds to DSM data. The second to last columns correspond to true DTM, proposed method, SMRF, gLiDAR and EMD-filtering results respectively. As can be seen in this figure, all other three methods in the literature had more problems in filtering large buildings. Detailed non-ground object detection results are also given in Figure 5.24. Here columns from left to right correspond to the proposed method, SMRF, gLiDAR and EMD-filtering results respectively. True non-ground object detections are labeled in green, true ground detections are labeled in gray, miss non-ground detections are labeled in blue, and false non-ground detections are labeled in red in these figures. The problem in detecting large buildings is clearly seen in this figure. All three methods in the literature have miss detections on large objects. However, the proposed method only has miss detection problems on objects which are connected to another higher object. This is because of the segmentation algorithm. At the end of segmentation, it is assumed that the object is taller then its immediate neighbors. Hence, it is tested whether the elevation height difference of the segmented object and its perimeter is above a threshold. This assumption may fail when buildings are connected side by side and one is taller than the other. Part of the largest building is lower than its surrounding for all directions in the Utah-1 test data. Hence, it could not be detected. The reason for this result can be explained as follows. The missing part did not get sufficient votes in the probabilistic voting step. Hence, no segmentation has started there. The SMRF and EMD methods also had a problem in detecting the same part of the building. gLiDAR did not detect most parts of the same building. There are also similar results on the Utah 7-8-9 test samples. Here, some buildings which are connected to a taller building could not be detected by the proposed method.

Object detection performance of the proposed and the other three methods are tabulated in Table 5.1. The values in this table are given in percentages. As can be seen in this table, the proposed method performed slightly worse than SMRF and EMD and better than gLiDAR in terms of all metrics on the Utah-1 test data. The proposed method performed better than the other three methods on the Utah-2 test data. Here, the other three methods have large Type-I error because of the miss detection of the largest building. All three methods have similar results on the Utah-3 test data. For Utah 4-6 test data, even though the terrain is sloped and the proposed method is not based on generated DTM, the obtained result is similar to the other methods. The proposed method has miss detections on the Utah-7 test data. Therefore, it has a high TI error. The proposed method has the best K, TE, TI and TII values on the Utah 8-9 test data. These results can be summarized as follows. If buildings are not large in the test data (as in Utah-3), all four methods have good and similar results. If there are large buildings in the test data (as in Utah-1-2-7-8-9), other methods mostly fail where the proposed method works properly. In a sloped terrain, SMRF has the best score with the proposed and gLiDAR methods having similar values. The proposed method outperforms other methods on detecting connected and large buildings.

Table 5.1. Performance results on the NSF Open Topography LiDAR data set.

Proposed Method					**SMRF [100]**				**gLiDAR [101]**				**EMD [127]**			
Test-Case	**K**	**TE**	**TI**	**TII**	**K**	**TE**	**TI**	**TII**	**K**	**TE**	**TI**	**TII**	**K**	**TE**	**TI**	**TII**
Utah-1	97.34	1.08	1.48	**0.02**	**97.86**	**0.87**	**1.19**	0.03	91.87	3.22	4.32	0.03	97.83	0.88	1.21	0.05
Utah-2	**98.59**	**0.65**	**1.01**	**0.03**	83.27	7.45	10.52	0.07	83.39	7.40	10.44	0.14	83.06	7.55	10.61	0.18
Utah-3	97.53	1.14	1.76	**0.03**	**99.36**	**0.30**	**0.45**	**0.03**	98.83	0.54	0.82	0.05	99.48	0.24	0.34	0.07
Utah-4	97.12	1.43	2.41	0.28	**97.97**	**1.01**	1.60	0.32	97.09	1.45	2.55	**0.13**	97.89	1.05	**1.04**	1.06
Utah-5	97.28	0.96	1.14	0.31	**98.35**	**0.58**	0.66	**0.30**	96.78	1.13	1.40	**0.18**	98.02	0.70	**0.53**	1.28
Utah-6	94.57	2.43	3.45	0.33	96.59	1.54	2.01	0.60	94.18	2.60	3.70	**0.32**	**97.06**	**1.33**	**1.48**	1.06
Utah-7	87.40	6.08	9.65	**0.09**	83.49	7.94	12.14	0.33	**87.49**	**6.05**	**9.52**	0.24	77.38	10.81	15.58	1.26
Utah-8	**97.21**	**0.97**	**1.24**	**0.02**	74.09	8.07	9.48	0.06	82.79	5.60	6.71	0.39	69.24	9.50	10.53	3.42
Utah-9	**88.78**	**4.85**	**6.81**	**0.07**	73.21	11.07	14.24	0.46	72.87	11.21	14.31	0.88	63.93	14.57	17.72	2.10
Mean	**95.09**	**2.18**	**3.22**	**0.13**	89.36	4.31	5.81	0.25	89.48	4.36	5.98	0.26	87.10	5.18	6.56	1.16
Min	**87.40**	0.65	1.01	**0.02**	73.21	0.30	0.45	**0.03**	72.87	0.54	0.82	**0.03**	63.93	**0.24**	**0.34**	0.05
Max	98.59	**6.08**	**9.65**	**0.33**	99.36	11.07	14.24	0.60	98.83	11.21	14.31	0.88	**99.48**	14.57	17.72	3.42
Std	**4.12**	**1.96**	**3.03**	**0.13**	10.87	4.23	5.65	0.21	8.57	3.51	4.61	0.26	14.01	5.46	7.06	1.09

Results in Table 5.1 can be summarized as follows. The proposed method has the best mean Kappa score with 94.91 % where SMRF, gLiDAR and EMD mean Kappa scores are 89.36 %, 89.48 % and 87.10 % respectively. The proposed method has the best mean TE, TI, and TII values with 2.25 %, 3.30 %, and 0.17 % respectively. It also has the lowest standard deviation (std) of all metrics. This shows the robustness of the proposed method on such a diverse test set.

5.2.6.3. WorldView Stereo Image based DSM Data Set

The second data set is composed of DSM generated from WorldView-2 stereo image pairs. More information on DSM generation from these can be found in [31, 134]. Spatial resolution of the available DSM is 0.5 m. In this test set, the sharpness of DSM is slightly worse compared to the first data set. This affects building boundaries such that they have smoother transitions towards the ground. Therefore, this data set shows how the proposed method works under these constraints.

In Figure 5.25 we give the WorldView data set and filtering results. Here columns from left to right correspond to panchromatic image, DSM, generated DTM and object filtering results respectively. True non-ground object detections are labeled in green, true ground detections are labeled in gray, miss non-ground detections are labeled in blue, and false non-ground detections are labeled in red in these figures. In the first column of Figure 5.25, four panchromatic test images where the first two are residential and last two are industrial regions in Istanbul, Turkey are provided. In the second column of the same figure, DSM data of the same locations are provided. In residential regions, buildings have regular size but closely located buildings look like connected due to the nature of the DSM data. In industrial regions, some buildings have very large footprints. In the third column, generated DTMs are given. In the last column, object detection results are given. Due to lack of ground truth information for object locations, they are extracted manually.

Quantitative results of the second data set are given in Table 5.2. The values in this table are given in percentages. Unfortunately, the other three methods could not be used on this data set since they need LiDAR points as input. The proposed method performs well in residential

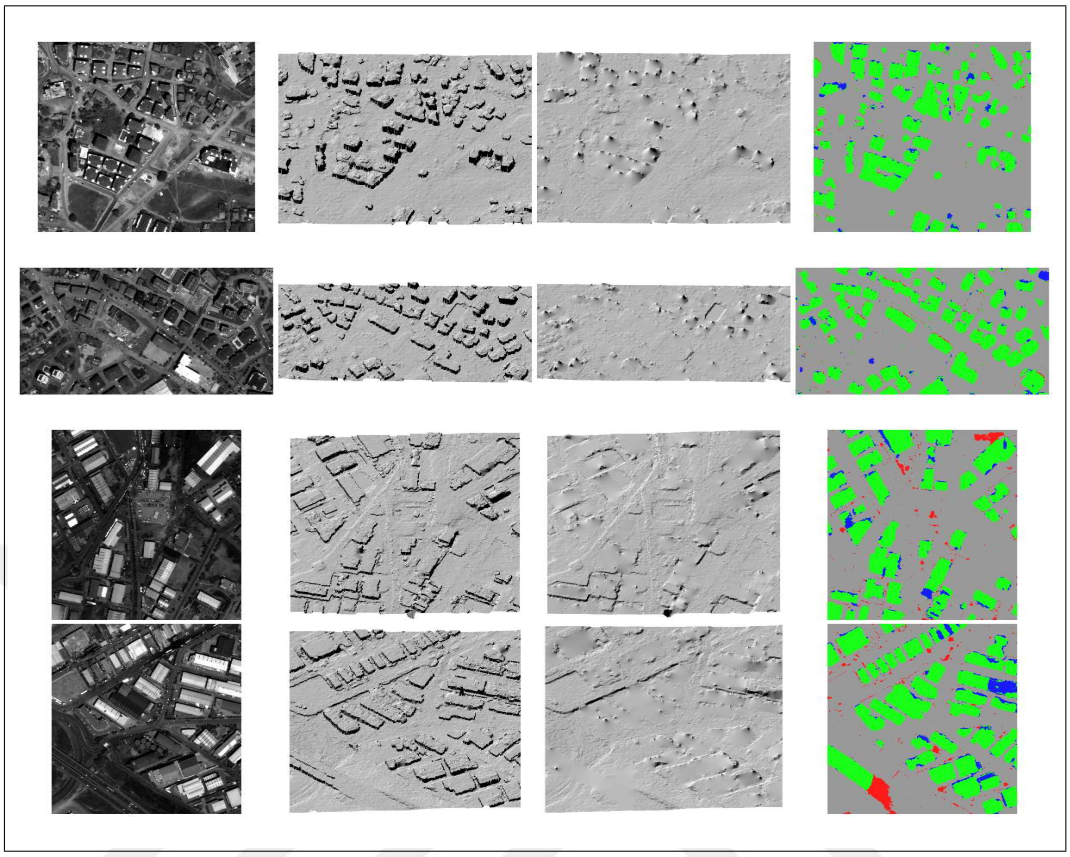


Figure 5.25. Filtering results on the WorldView stereo image based DSM data set.

regions where the Kappa score for the first two data set is over 85 %. In industrial regions, Kappa scores are worse. False detections cause large TII error in these test data. There are some embankments where one side of them looks like an object and the other side looks like belonging to the terrain in the Industrial-1 test data. Thus, it misguides the segmentation process and causes false detection (in red at the upper right corner of the image). There is a bridge in the Industrial-2 test data which is correctly detected as an object. However, the terrain which is connected to the bridge was also detected as an object, which caused a large TII error. In the upper right of the same test data, part of a large building could not be detected. Even though the building has a flat roof, it has an irregular distribution of height data on its roof. Hence, the segmentation method could not work properly.

5.2.6.4. ISPRS Vaihingen Data Set

The third data set is from ISPRS. They provide a semantically labeled data set which holds LiDAR data for several benchmark test locations [135]. In this study, the first five tiles of Vaihingen data are used. For this data set, trees and vegetation are removed by using an NDVI mask from object detection results. Therefore, it is assumed that the remaining object detection results represent buildings. Hence, the performance of the proposed method was evaluated on building detection results of the ISPRS reference data. To be compatible with other methods using this data set, the completeness (CP), correctness (CR), and F1 score results are given in Table 5.3 [135]. Here, pixel wise building detection results are provided. As can be seen in this table, mean completeness and correctness scores are close where mean completeness is 90.33 % and mean correctness is 88.92 %. The F1 score, which is harmonic mean of completeness and correctness, is 89.57 %. The reader can check the most recent comparison results from the mentioned ISPRS website [135]. To note here, most of the other methods use supervised approaches in the ISPRS data. Although the proposed method has an unsupervised approach, it still provides good results.

Building detection results on three test samples from the third data set are given in Figure 5.26. Here columns from left to right correspond to the 2D image, DSM, and building detection results respectively. In the last column, true non-ground object detections are labeled in green, true ground detections are labeled in gray, miss non-ground detections are labeled in blue, and false non-ground detections are labeled in red. As can be seen in this figure, most of the

Table 5.2. Performance results on the WorldView DSM data set.

Test-Case	**K**	**TE**	**TI**	**TII**
Residential-1	86.71	4.64	5.26	2.24
Residential-2	86.28	5.27	6.44	1.49
Industrial-1	83.44	6.45	6.12	7.40
Industrial-2	79.25	9.18	7.71	12.24
Mean	83.92	6.39	6.38	5.84

buildings are detected. In cases where a building is partly surrounded by taller buildings or trees, some miss detections occur. Such a case can be seen in the last row of Figure 5.26 where a building is surrounded by trees at the bottom center of the test data. Hence, it could not be detected.

5.3. EMD BASED DSM FILTERING

In the previous section, we proposed a segmentation method for DSM and LiDAR data using the voting process and a novel morphology based segmentation algorithm. Most of the proposed methods in the literature perform well in regions where the terrain is flat and non-ground objects have regular shape. However, for steep sloped and abruptly changing regions, they could not perform as expected. Morphology based methods seem adaptable to steep sloped regions if the structuring element size is chosen well. Though it requires a pre-knowledge of the terrain. This prevents its usage for all region types automatically. Besides, detecting large and small objects at the same time are always hard while using morphology based methods.

To overcome the mentioned problems, a novel method using Empirical Mode Decomposition (EMD) is proposed. EMD has been used successfully by the remote sensing community for hyperspectral image classification. Demir and Erturk [136] used EMD to decompose hyperspectral image bands. They used the sum of low order IMFs as features in SVM

Table 5.3. Performance on the ISPRS Vaihingen, Germany data set.

Test-Case	**CP**	**CR**	**F1**
Vaihingen-1	93.63	85.78	89.53
Vaihingen-2	88.30	89.40	88.80
Vaihingen-3	89.41	87.72	88.55
Vaihingen-4	89.10	91.00	90.00
Vaihingen-5	91.22	90.69	90.95
Mean	90.33	88.92	89.57

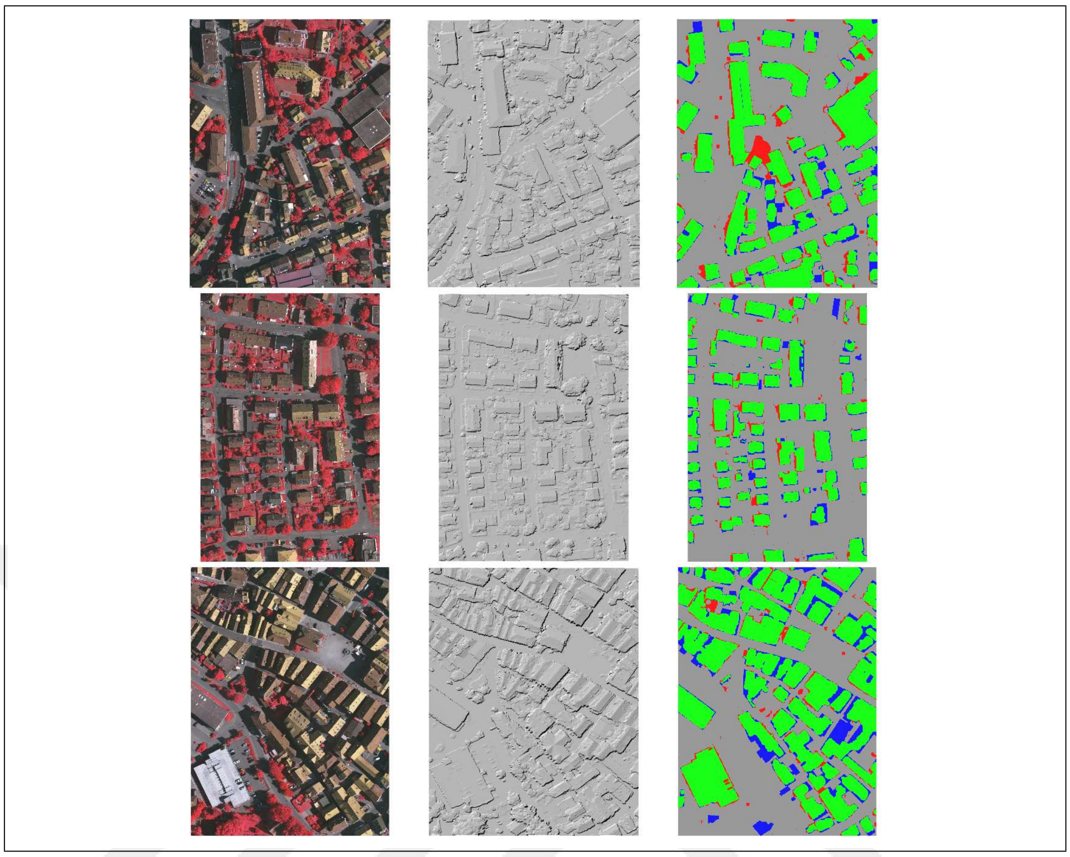


Figure 5.26. Building detection results on the Vaihingen 1-3-5 of ISPRS data set.

for classification. In a similar study, Demir *et al.* [137] used two-dimensional EMD for hyperspectral image classification. Gormus *et al.* [138] used EMD and wavelets together for dimensionality reduction in hyperspectral images. Erturk *et al.* [139] applied EMD to each hyperspectral image band to generate new features for classification purposes. This way, they improved the hyperspectral image classification accuracy significantly. He *et al.* [140] used EMD and the morphological wavelet transform to extract features for hyperspectral image classification.

The proposed method is based on iteratively applying two-dimensional EMD to the given LiDAR data in Digital Surface Model (DSM) form. DSM is generated from LiDAR point data using nearest neighbor interpolation. The local, nonlinear and non-stationary characteristics of EMD may help to solve the DTM generation and point classification (as ground and above-ground objects) problems in a successful manner.

Within the proposed method, EMD is used in an iterative manner. For each EMD iteration, the corresponding residual signal is obtained and it is used as an adaptive threshold to detect above-ground objects crudely. Then, the obtained results are merged using binary logical or operation. This step is explained in Section 5.3.4. Then, detected above-ground objects are discarded from DSM data and image inpainting is applied to construct the final DTM. Using it and slope based thresholding, above-ground objects can be detected in a fine manner. This part is explained in Section 5.3.6. To explain these steps visually, flowchart of the proposed method is proposed in Figure 5.27. The proposed method is tested on two publicly available data sets. Besides, it is compared with other methods in the literature. The obtained results are provided in Section 5.3.8.

5.3.1. Empirical Mode Decomposition

Empirical Mode Decomposition (EMD) is a data-driven and model free method used to decompose a signal into additive and finite set of oscillatory modes [141]. EMD is based on the local characteristics of the signal. Therefore, it can be applied to nonlinear and non-stationary signals such as the ones in this study.

EMD decomposition of a two-dimensional signal $s(i,j)$ (with horizontal and vertical indices as i and j) can be represented as

$$s(i,j) = \sum_{k=1}^{K-1} m_k(i,j) + r(i,j) \tag{5.5}$$

where, $m_k(i,j)$ are the Intrinsic Mode Functions (IMF) and $r(i,j)$ is the residual signal. Here, $K-1$ is the number of IMFs in EMD. When all the IMFs and the residual signal is summed, the original signal is obtained [141]. In this decomposition, the first IMF corresponds to the fastest fluctuating part of the signal. The residue represents the slowest fluctuating part of the signal. In this study, the residue is used for DTM generation.

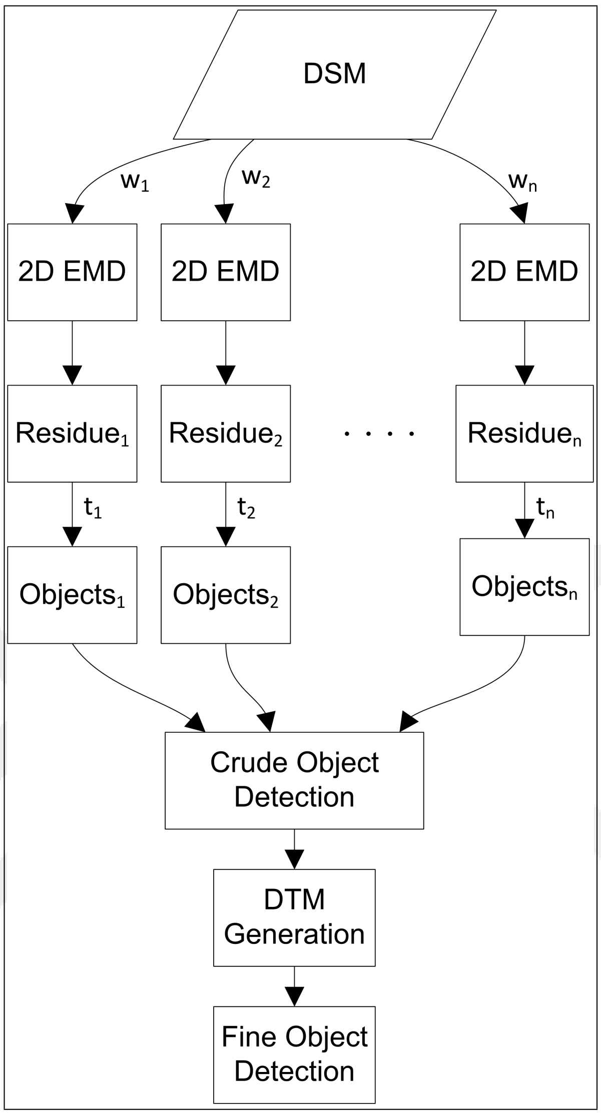


Figure 5.27. Flowchart of the proposed EMD based DSM filtering method.

5.3.2. IMF Extraction

IMF extraction is based on an iterative process called as sifting [141]. Within this method, local maxima and minima of the signal are extracted. Then, they are used for constructing upper and lower envelopes respectively. Based on these, average of the envelopes is calculated. This value is subtracted from the original signal. The result is checked whether it satisfies being an IMF. If this condition is satisfied, then the result is kept and sifting continues on the average of the envelopes. Otherwise, sifting continues on the subtracted result.

Extracted IMFs have specific characteristics as follows. First, the number of local extrema and zero crossings of each IMF must vary by at most one. Second, the average of upper and lower envelopes of each IMF should be zero. Third, the number of zero crossings in the IMF decreases with respect to its index value k. The IMF extraction algorithm based on these constraints are provided in Algorithm 5.3. As this algorithm is run, IMF components and the residue term is obtained.

Algorithm 5.3.IMF extraction algorithm based on iterative sifting process

```
BEGIN
$k$; iteration number
$k = 1$
$s(i, j)$; image to be decomposed
$r(i, j) = s(i, j)$
repeat
  extract local maxima and minima of $r(i, j)$
  interpolate local maxima to obtain $e_u(i, j)$
  interpolate local minima to obtain $e_l(i, j)$
  $a(i, j) = (e_u(i, j) + e_l(i, j))/2$; average of envelopes
  $d(i, j) = s(i, j) - a(i, j)$; oscillating part of the image
  if $d(i, j)$ satisfies being IMF conditions then
    $m_k(i, j) = d(i, j)$
    $r(i, j) = a(i, j)$
    $k = k + 1$
  else
    $r(i, j) = d(i, j)$
  end if
until less than two local maxima or minima in $r(i, j)$
return $m_1(i, j), m_2(i, j), \cdots, m_{K-1}(i, j), r(i, j)$
STATE END
```

5.3.2.1. Extrema Detection

In an image, local extrema (maxima or minima) may be defined in several ways. Generally speaking, a pixel is a local maximum if it's value is higher than all its neighbors. There are also other definitions such as regional extrema, extrema in a neighboring window, and extrema extraction with morphological reconstruction [142, 143].

In this study, an extrema detection method is proposed based on morphological opening with varying structuring element size, w_i. In the proposed method, a local maximum may be the entire top of an object. Hence, if subtraction of the opening result from original DSM (tophat transform result) is larger than a threshold, then it is accepted as a local maximum and all other pixels are assumed to be local minima [125]. This way, any pixel in the image will be either local minima or maxima.

5.3.2.2. Envelope Estimation

In sifting process, upper and lower envelopes are estimated from local maxima and minima respectively. Typically a cubic interpolator is used for envelope estimation. There are also other interpolator types used in two-dimensional EMD applications [142, 144, 145]. In this study, the upper and lower envelopes are formed using local maxima and minima extracted in the previous section. To extract the upper envelope, local maxima pixels in the image are labeled as missing. Using the modified image, the upper envelope is estimated by image inpainting. To extract the lower envelope, local minima pixels in the image are labeled as missing. Using the modified image, the lower envelope is estimated by image inpainting. Here, Derrico's image inpainting implementation is used [129]. This envelope detection method gives better upper and lower envelopes when there are small number of maxima or minima in the image.

5.3.3. Applying EMD on a Sample DSM Test Data

A sample DSM test data (Utah3) is picked to demonstrate how the proposed method works. This test data is obtained from the NSF web site as explained in detail in Section 5.3.8.2. The Utah3 DSM test data is provided in Figure 5.28. In this data, there are buildings and vegetation on a hillside. To explain the method in detail, a vertical profile of the DSM (with line number 650) will be used. This profile is plotted in Figure 5.29.

First, EMD is applied to the Utah3 test data. To do so, one needs to define structuring element sizes for extrema detection. As explained in Section 5.3.2.1, a morphological tophat transform is applied on DSM and the result is thresholded for each element size. Pixels exceeding the

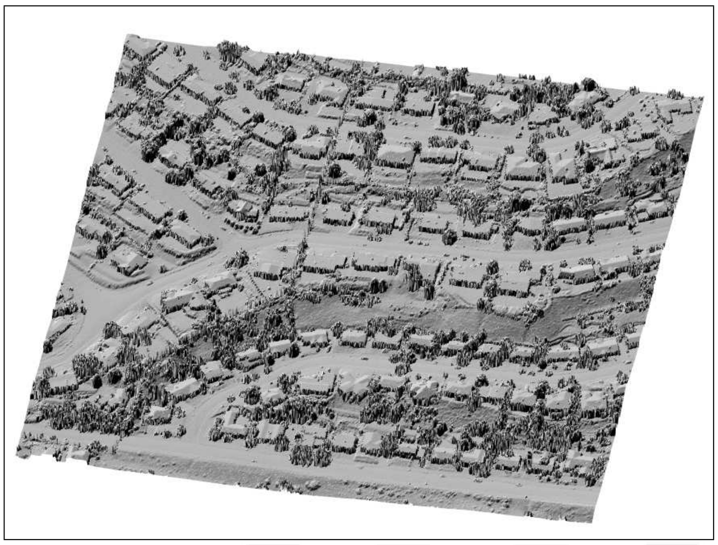


Figure 5.28. The Utah3 DSM test data.

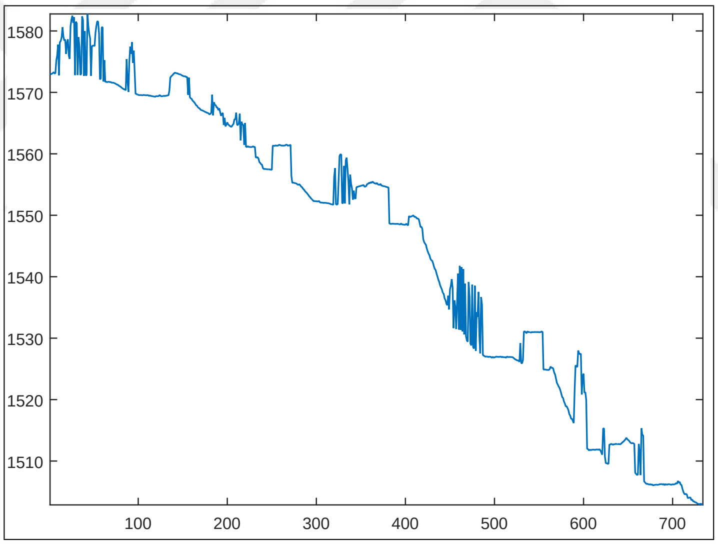


Figure 5.29. Vertical profile of the Utah3 DSM test data with line number 650.

threshold are considered as local maxima and the others as local minima. For a small element size, large buildings wont exceed the threshold and wont be used as local maxima. Hence, the estimated residue will be passing over those buildings. For a better understanding, two small and two larger structuring element sizes, w_i, are used as 1, 3, 8 and 18 m on the one dimensional profile given in Figure 5.29. Here, these structuring element sizes are picked to span a wide operation range. The rationale in selecting these parameters will be explained in detail in Section 5.3.8. The extracted residues are given in Figure 5.30. As can be seen in

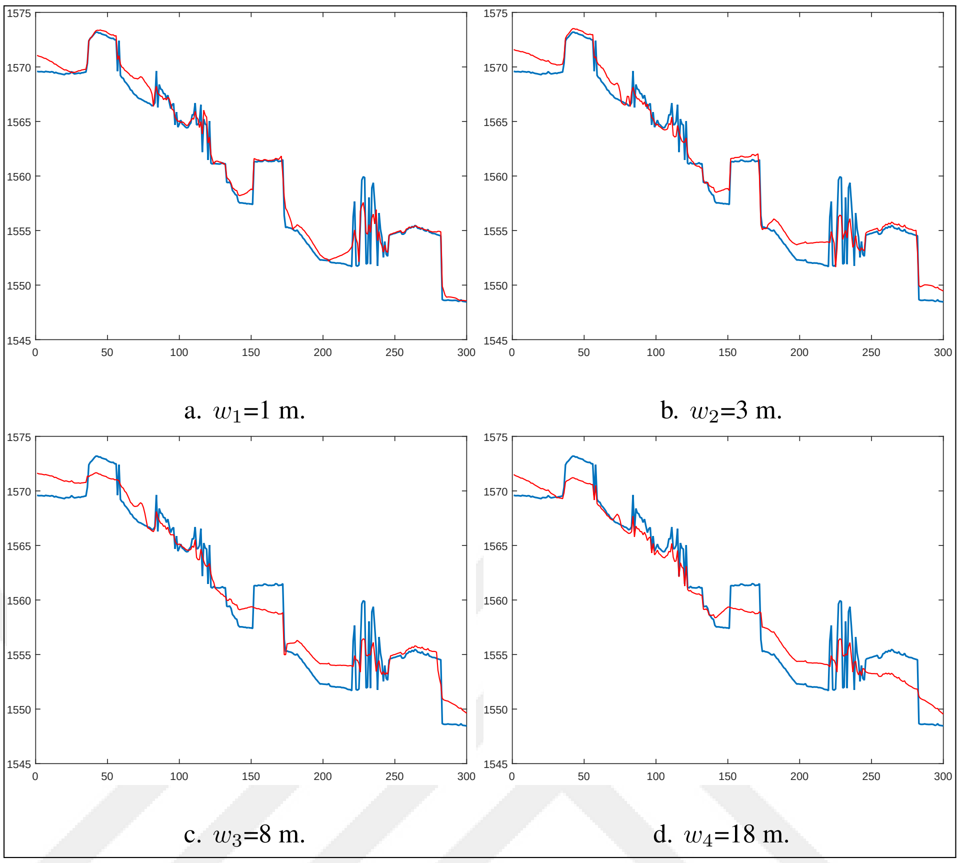


Figure 5.30. Residues obtained from the vertical profile of the Utah3 test data.

this figure, obtained residues hold less or more detail about the DSM based on the selected structuring element size. To be more specific, as the disk size increases, the extracted residue can track the terrain profile better.

5.3.4. Iterative EMD based Crude Object Detection

The residual signal can be used for above ground object detection from DSM data. Unfortunately, one residue is not enough to detect all objects (with varying size) in DSM data. Therefore, EMD is used in an iterative manner as explained next.

The residual signal is a good candidate of being an adaptive threshold for above ground object detection in DSM data. Although there are other adaptive thresholding methods in the literature, EMD characteristics (locality, nonlinearity, and non-stationarity) provide a much better threshold value. Hence, above ground objects, O_i, can be detected in a crude manner as

$$O_i = (Z - R_i) \geq t_i \tag{5.6}$$

where Z is the elevation value in DSM, R_i is the residue term, and t_i is the threshold value. Detected objects are called as crude since these do not represent the final (and fine) detection results.

The residue term depends on the structuring element size, w_i, as mentioned in Section 5.3.2.1. Changing this value allows reforming the residue term such that objects with different size can be detected. Therefore, several structuring elements can be used with different size for extrema detection in a parallel manner. Since each generated residue will have different characteristics, the corresponding threshold should also be set accordingly. Therefore, a different threshold, t_i, is used for each residue.

Next, the iterative residue extraction and thresholding process is applied to the Utah3 test data. In the previous section, residues were obtained with different structuring element sizes. Using these, above-ground objects can be extracted by applying the threshold value, t_i, as 0.1, 0.5, 1.3, 2.9 m. The rationale in selecting these values will be explained in Section 5.3.8. Crude object detection results for the Utah3 test data are given in Figure 5.31. As can be seen in this figure, using different residues lead to different object detection results.

5.3.5. Combining Object Detection Results

As mentioned in the previous section, applying EMD in an iterative manner leads to object detection results with different sizes. These can be combined by a logical or operation [126]. The rationale here is as follows. The logical or operation gives a detection result even if the detection is done in one scale. Hence, the combined crude object detection will be as

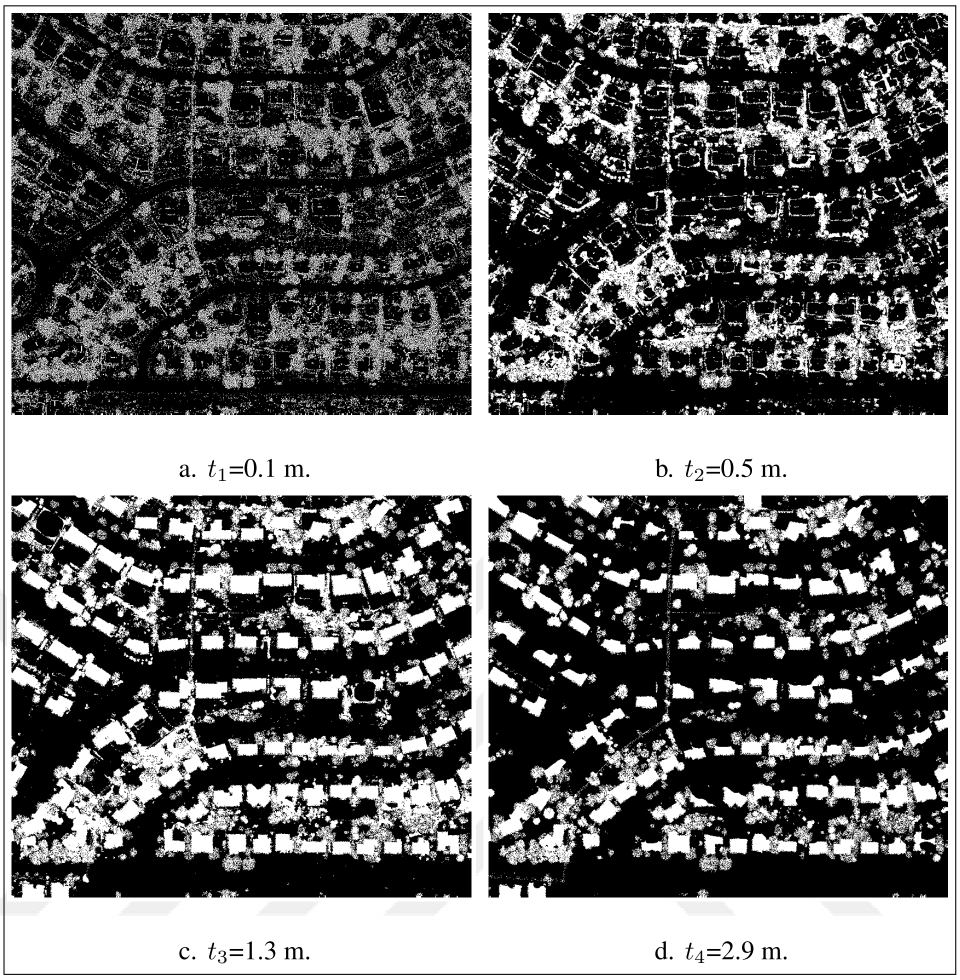

a. t_1=0.1 m.
b. t_2=0.5 m.
c. t_3=1.3 m.
d. t_4=2.9 m.

Figure 5.31. Crude object detection results from the first to the fourth residue.

$$O = \bigcup_{l=1}^{L} O_l \tag{5.7}$$

where $\bigcup$ represents the setwise logical or operation. O_l for $l = 1, \cdots, L$ represent crude object detection results. Combined object detection result for the Utah3 test data are given in Figure 5.32. As can be seen in this figure, combining the result provides good above ground object detection for the Utah3 test data.

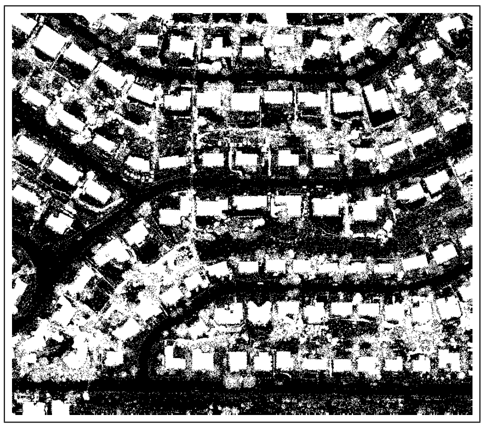


Figure 5.32. Combined crude object detection result for the Utah3 test data.

5.3.6. DTM Generation and Fine Object Detection

Crude object detection results can be used in DTM generation. This leads to fine object detection. In this section, both methods are explained.

5.3.6.1. DTM Generation

In general, ground points will be needed to generate the DTM of a region. To do so, non-ground points should be generated. In this study, image inpainting is used for this purpose [128]. Here, non-ground pixels (extracted in the previous section) are filled with the inpainting process. Hence, DTM is generated.

The generated DTM from the Utah3 test data is provided in Figure 5.33.a. The ground truth DTM (downloaded from the NSF Open Topography website) is provided in Figure 5.33.b [124]. More information on this data is given in Section 5.3.8.2. As can be seen in Figure 5.33, the ground truth and generated DTMs are similar.

5.3.6.2. Fine Object Detection

If the generated DTM perfectly matches to the original DTM, then anything above the ground may be labeled as an object. Hence, anything above ground is called as a non-ground object.

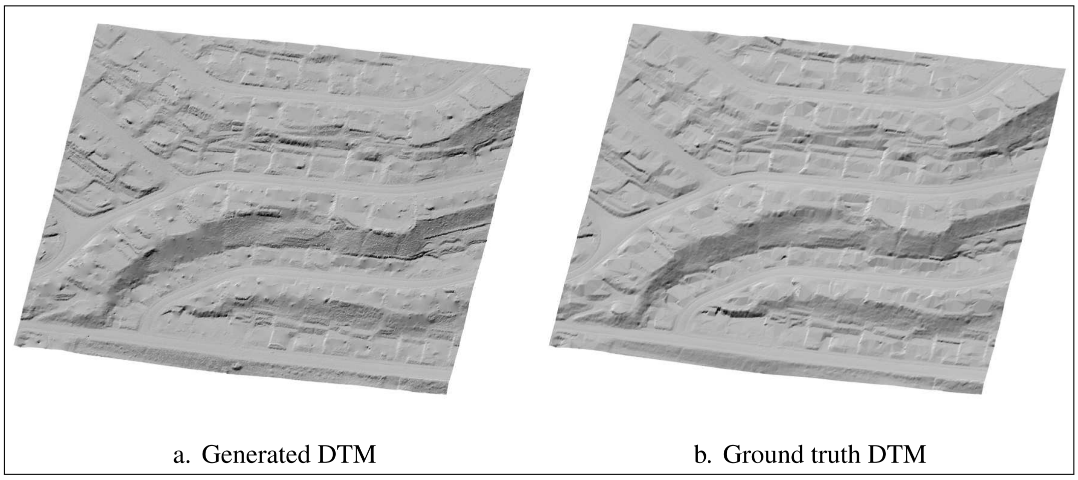


Figure 5.33. DTM generation result on the Utah3 test data.

To detect these, the generated DTM is subtracted from the original DSM and the normalized DSM is obtained. In this form, an elevation threshold can be used to detect non-ground objects. However, this may not be a good option to detect objects for hilly or steep sloped regions since in these regions generating a good DTM is not easy. Such examples are provided in Section 5.3.8. Therefore, slope based thresholding is used here initially proposed by Pingel *et al.* [100].

This method is modified by taking the squared power of the slope of DTM and adding it to a fixed threshold. For flat regions, the slope will be small. Hence, taking its power will not change the threshold much. For steep sloped regions, the slope will be large. Therefore, it will increase the threshold and decrease false detection results. Based on these observations, the fine object detection, FO, will be as

$$FO = Z_n \geq (\rho + \mathit{\Delta}^2) \tag{5.8}$$

where Z_n is the normalized DSM. ρ is the fixed elevation threshold value and $\mathit{\Delta}$ is the slope at given location. Actual values for these parameters are obtained experimentally as explained

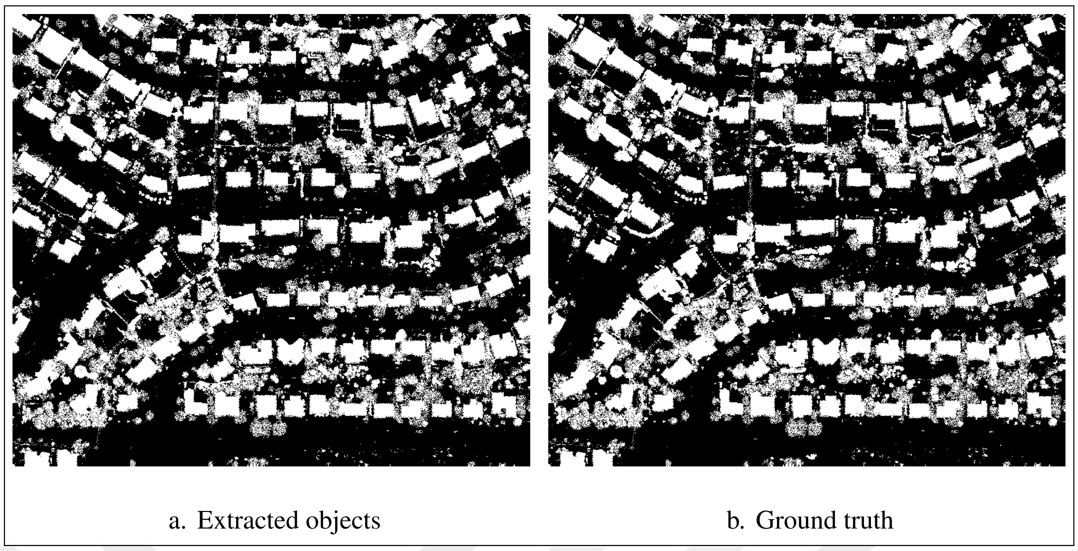


Figure 5.34. Fine object detection results on the Utah3 test data.

in Section 5.3.8. This step is called as fine object detection, since it benefits from both crude object detection and the DTM generation results.

Fine object detection results for the Utah3 test data are provided in Figure 5.34. In the same figure, the ground truth data is provided as well. As can be seen in this figure, fine object detection works fairly well. Quantitative comparison results will be provided in Section 5.3.8.2.

5.3.7. Pseudo Code of the Proposed Method

In order to clarify the above procedure, a different approach is provided here. Hence, the pseudo code is given in Algorithm 5.4. Here, the proposed method is explained using abstract operations. Hence, no specific programming language is targeted. We believe this will help the reader to grasp the proposed method better.

Algorithm 5.4. Pseudo code of the proposed EMD based filtering method

```
BEGIN
$Objects$; initial binary object class
$I_s$; corresponding DSM
$I_d$; generated DTM
$\rho$; elevation threshold in meters
$N$; interval number
$w_i$; disk size in meters
$t_i$; threshold in meters
$i = 1$; iteration number
$\sim$; complement
repeat
   $residue = EMD(I_s, t_i, w_i)$;
   $crudeObjects = (I_s - residue) > (t_i * 0.75)$;
   $Objects = Objects \mid crudeObjects$;
   $i = i + 1$;
until i>N
$I_d = inpaint(\sim Objects, I_s)$;
$fineObjects = (I_s - I_d) > (\rho + \Delta^2)$;
return $I_d, fineObjects$
END
```

5.3.8. Experiments on 3D Data

The performance of the proposed method is tested on two publicly available data sets. The first data set is obtained from ISPRS. Although this is a relatively old data set, it has been used as a benchmark on several studies. This data set contains various terrain characteristics like vegetation, building, road, railroad, bridge, and water surface. The second data set is obtained from the publicly available NSF Open Topography website. Forest and residential regions are specifically picked for this data set. For both data sets, the performance of the proposed method is provided in terms of object detection. Here, Kappa scores are used [130]. We also compare the proposed method with other methods in the literature. Based on these comparisons, the strengths and weaknesses of the proposed method are emphasized. MATLAB codes used in experiments are also provided in [146]. The reader can also evaluate our results using these codes.

5.3.8.1. ISPRS LiDAR Data Set

The first data set used in this study is provided by Sithole and Vosselman [83]. It is publicly available at [117]. This data set is composed of 15 airborne LiDAR samples of which the first nine are from urban regions. The remaining six samples are from rural regions. The point spacing in urban areas (first nine data set samples) is between 1 and 1.5 m. For rural areas (last six data set samples) the point spacing is between 2 and 3.5 m. Sithole and Vosselman specifically picked these 15 samples such that they should be challenging for automated methods. The ground truth data is also provided for this data set. Here, each LiDAR point is manually classified either as ground or above ground object.

The performance of the proposed method is compared with five other (best performing) methods in the literature. The comparison results are tabulated in Table 5.4 in terms of Kappa score percentages. For the proposed method, 10 iterations are applied with the structuring element size, w_i, set from 1 to 20 m in 10 equally spaced values. The corresponding object detection threshold values are set from 0.1 to 4 m in 10 equally spaced values. In selecting these values, the rationale was as follows. The maximum structuring element size is selected in order to filter out the largest object in the tophat transform for the EMD extraction step. Hence, it can be used as local maxima in EMD extraction. Object detection threshold values are set according to the structuring element size such that most non-ground objects can be detected. The iteration number does not affect the performance much. Such a test is provided in Figure 5.37.a. There, the effect of other parameters on the performance is also analyzed. In Eqn. 5.8 the elevation threshold is selected as $\rho = 0.6$m.

As can be seen in Table 5.4, the proposed method has the best mean Kappa score. It also has the lowest standard deviation for Kappa scores. The standard deviation indicates the reliability of the obtained mean Kappa score. It also indicates the performance of the proposed method on different data sets as long as the mean Kappa score is high. It can be said that with such a high mean Kappa score and low standard deviation, the proposed method performs fairly well on different data samples compared to other methods in the literature. To briefly summarize some data characteristics, samp51 has slope with vegetation. Samp52 and samp53 has very steep sloped regions. Samp54 has dense ground cover. Samp61 has large gaps with embankments in the road. Samp71 has a bridge with underpass and road with embankments. In five samples out of 15 the proposed method has the best Kappa score. Notice that, for samp53 and samp61 the proposed method performs fairly well compared to the ones in the literature. In these test samples, the DSM has steep slopes and sharp ridges. These results indicate that, the proposed method can handle such region types fairly well. The proposed method also has the best Kappa score for samp41. In this test sample, there are very large objects. Here, most morphology based methods failed to detect these large objects since they need manually adjusted parameters in morphological operations. On the other hand,

Table 5.4. Comparison with five other methods in literature on the ISPRS data set.

	Pfeifer [147]	**Axelsson [148]**	**Meng [149]**	**Pingel [100]**	**Hu [96]**	**Proposed**
samp11	66.09	78.48	70.96	82.4	**82.78**	82.04
samp12	91.00	93.51	92.28	93.8	**94.02**	93.77
samp21	92.51	86.34	93.79	94.43	94.26	**95.86**
samp22	84.68	91.33	87.83	**92.07**	91.76	89.56
samp23	83.59	**91.97**	83.35	87.02	90.47	89.45
samp24	78.43	88.5	82.83	89.49	**89.52**	88.96
samp31	**96.37**	90.43	93.31	95.00	93.41	92.24
samp41	78.51	72.21	88.27	78.41	87.47	**91.04**
samp42	93.67	96.15	**97.18**	93.07	97.10	89.84
samp51	89.61	91.68	81.18	90.74	91.49	**92.20**
samp52	41.02	83.63	58.43	78.8	**83.69**	83.15
samp53	30.83	39.13	25.60	47.24	53.06	**61.15**
samp54	88.93	93.52	80.61	92.65	**94.57**	92.48
samp61	47.09	74.52	50.16	75.38	71.08	**76.08**
samp71	66.75	**91.44**	64.11	90.52	90.54	89.90
Mean	75.27	84.19	76.66	85.40	87.01	**87.18**
Std	20.70	14.39	19.72	12.34	11.37	**8.80**

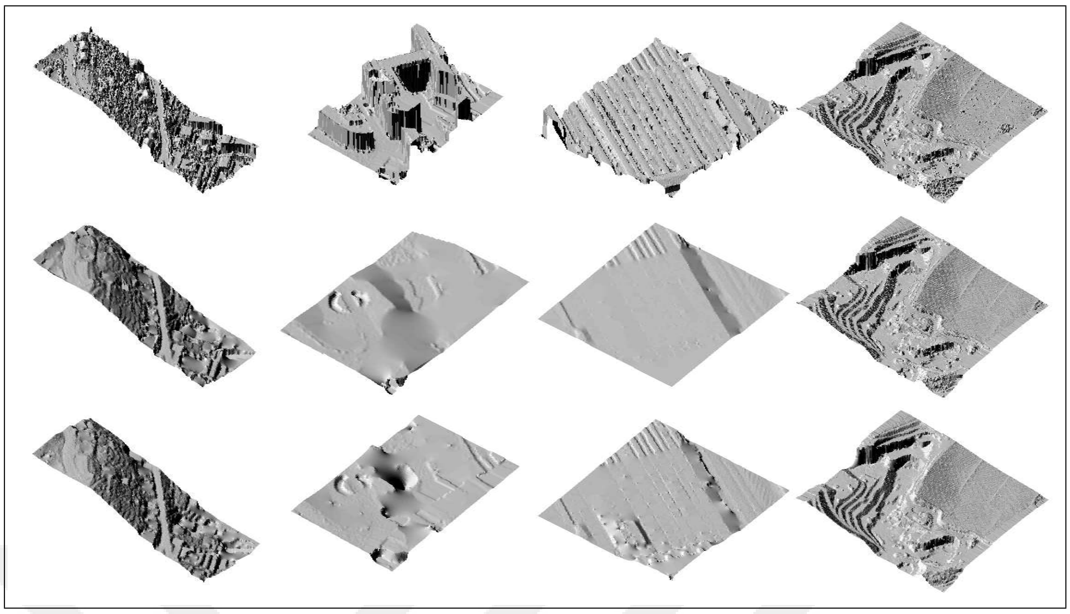


Figure 5.35. Visual DTM generation results for some ISPRS data samples.

the proposed method did not seem to have such a problem. The proposed method could not perform well on samp42 compared to the other methods. One possible reason for this poor performance is the structure of the data. Here, there is a railroad and most test points belong to one large object. Therefore, iterative EMD could not model this terrain characteristics well. For the remaining test samples in this data set, the proposed method has either the second or third best performance. These results indicate that the proposed method performs fairly well on the publicly available ISPRS data set compared to other five (best performing) methods in the literature.

To visually explain the obtained results, samp11, samp41, samp42, and samp53 are picked. As mentioned before, these are challenging test samples. The original DSM, generated and the ground truth DTMs for these test data are provided in Figure 5.35. In this figure, the first row represents the original DSM for samp11, samp41, samp42, and samp53 respectively. The second row represents the ground truth DTMs. The third row represents the generated DTMs by the proposed method. For the same data set, object detection results and the ground truth are provided in Figure 5.36. In this figure, the first row represents the ground truth for the samp11, samp41, samp42, and samp53 respectively. The second row represents the object detection results by the proposed method.

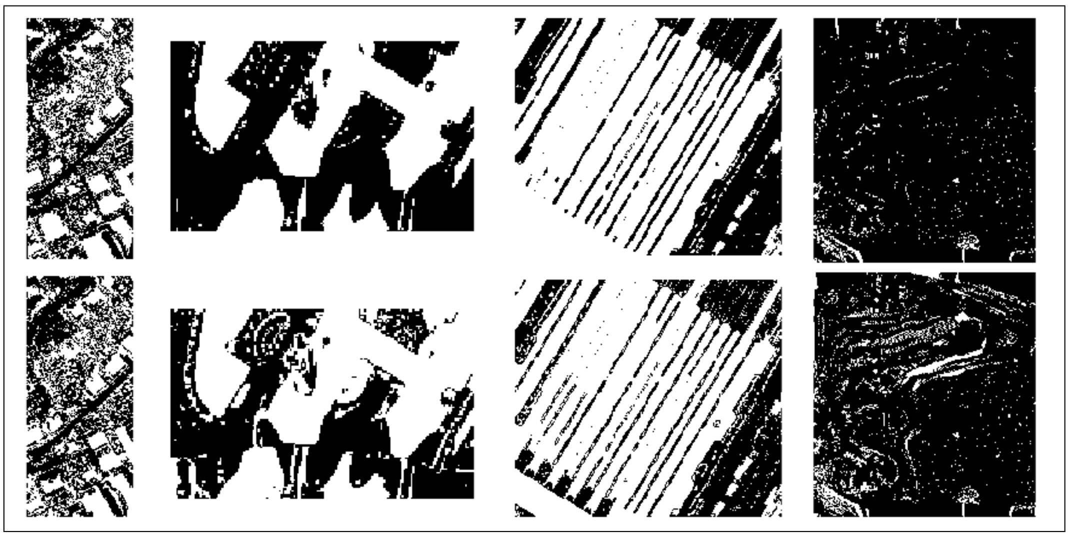


Figure 5.36. Visual object detection results for some ISPRS data samples.

In obtaining object detection performances, some parameter values have been set. In this section, the sensitivity of the proposed method to these parameter changes are analyzed. First, the interval number (N) is considered. In the previous section, this value is set as 10. In order to test the effect of this interval number on object detection performance, the following test is applied. The interval number is changed from 2 to 20 (while keeping the lower and upper limits of w_i the same) and the mean Kappa scores (in terms of percentages) on 15 ISPRS test samples is provided in Figure 5.37.a. As can be seen in this figure, the mean Kappa score increases fairly fast till the interval value reaches six. Afterwards, increase in the Kappa score becomes insignificant. This result can be explained as follows. The interval number directly affects the number of window size (w_i) used in operation. For example, when the interval number is four, w_i values will be as 1, 7, 14, and 20; keeping the lower and upper limits as 1 and 20. Each window size can be used to detect objects with a different size. Therefore, when the interval number is increased, more objects can be detected. To be on the safe side, the interval number is picked as 10 in this study.

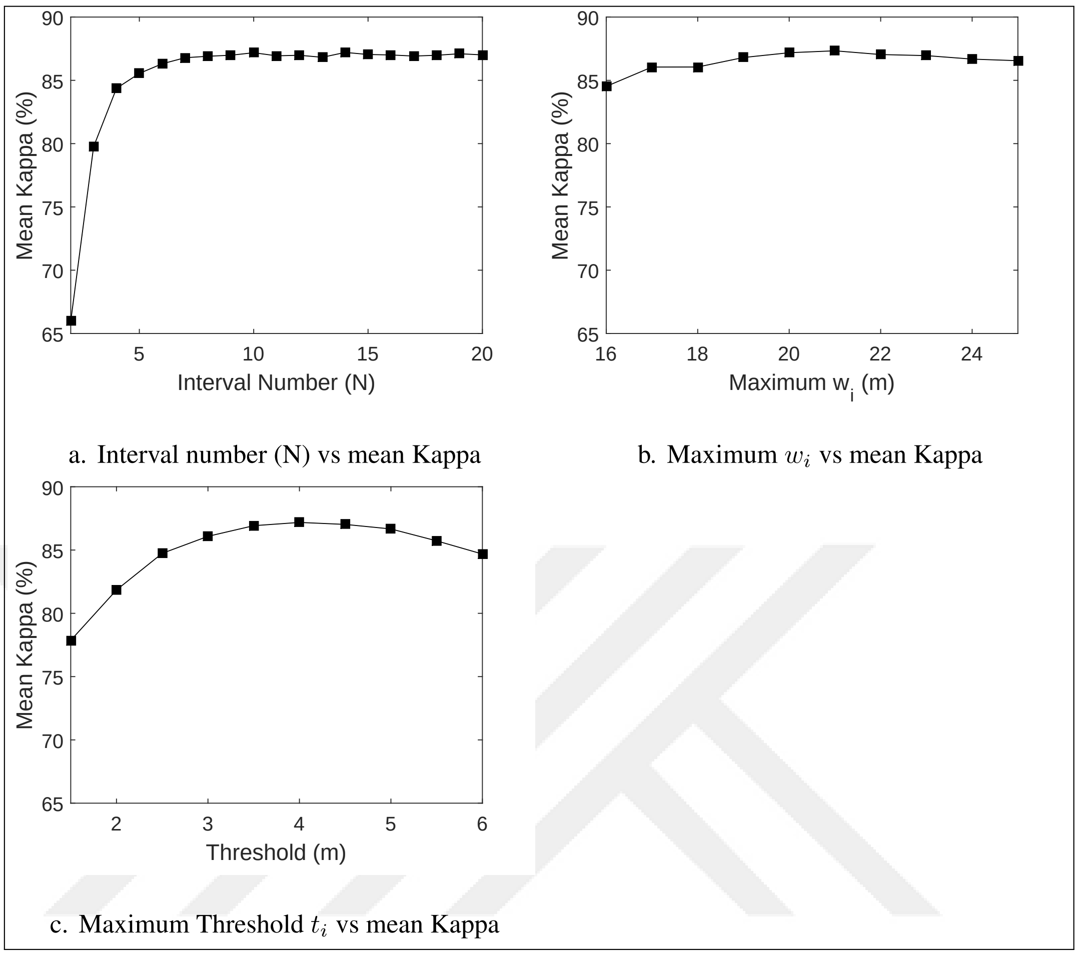
Next, the maximum w_i value in iterations is considered. The mean Kappa scores (in terms of percentages) on 15 ISPRS test samples is provided in Figure 5.37.b. Here, the change of the mean Kappa score of fifteen samples is provided when the maximum w_i value changes. Keeping the lower limit of w_i as 1, the maximum (upper limit) w_i value is varied from 16 to 25 m with 1 m steps. For all maximum w_i values, the interval number N is set as 10. To explain this figure better, if the maximum w_i is set as 16, this means that w_i values from 1 to 16 in 10 equally spaced values are used. These are rounded as [1, 3, 5, 7, 9, 12, 14, 16, 18, 20] in operation. As can be seen in Figure 5.37.b, when the maximum w_i value changes, the mean Kappa scores of fifteen samples do not change significantly. Only a 3% to 4% change in mean Kappa score is observed. Hence, the proposed method is almost insensitive to the maximum w_i value. Finally, the effect of the maximum threshold value is tested in Figure 5.37.c. To explain this figure better, if the maximum threshold is set as 4 m, this means that we use 10 equally spaced threshold values as [0.10, 0.53, 0.97, 1.40, 1.83, 2.27, 2.70, 3.13, 3.57, 4.0] m in operation.

Analysis of the plots in Figure 5.37 reveal that in order to get a mean Kappa score above 85% the maximum structuring element size should be typically around 20 m. The maximum threshold should be in the interval of 3 to 5 m and the interval number should be greater than four. These settings are considered as default throughout the study. If the method will be used only on urban area where there are very large objects (such as buildings) and the terrain is not sloped, the maximum element size may be increased and the maximum threshold may be decreased for better performance.

5.3.8.2. NSF Open Topography LiDAR Data Set

The second data set used in this study is obtained from the publicly available NSF Open Topography website [124]. From this website, the reader can download LiDAR point cloud data by defining an area of interest. The reader can also download the DTM of the selected area from the mentioned website. Here, the DTM is generated from LiDAR points in which data is stored in TIN structure [150].

Seven test samples are downloaded from the NSF Open Topography website. Three of these are from the HJ Andrews Experimental Forest. These test samples represent forest regions with dense trees. Four of the remaining test samples are from the State of Utah. These are from residential regions with houses and other non-ground objects nearby. The grid resolution of the test samples is 0.5 m. DTM of all test samples are also donwloaded. Due to lack of



a. Interval number (N) vs mean Kappa

b. Maximum w_i vs mean Kappa

c. Maximum Threshold t_i vs mean Kappa

Figure 5.37. The effect of parameters on the performance for the ISPRS data set.

manually generated ground truth for the test samples, the downloaded DTM is assumed as ground truth. Therefore, the provided results in this section will be relative to the downloaded DTM. Metadata for the NSF Open Topography data set is provided in Table 5.5.

To compare the performance of the proposed method, two publicly available methods are picked. The first method is by Pingel *et al.* [100]. MATLAB code for this method is available at [132]. In accordance with the author, this method is called as Simple Morphological Filter (SMRF). The second method is by Mongus *et al.* [101]. The software for this method is available at [133]. It is called as gLiDAR in accordance with the website. These two methods are specifically picked, since their performance was better than the others in the literature on the ISPRS data set. Besides, their software was available online. Hence, the reader can test them on a different data set for comparison purposes.

First, the proposed method is tested on forest region samples (Andrews1, Andrews2, and Andrews3). For the SMRF method, default parameter values are used as suggested at [132]. Within the gLiDAR software, the default parameter values are used suggested in [101]. For the proposed method, the same parameter settings are used as in Section 5.3.8.1. The DSM, ground truth DTM, and generated DTMs are provided in Figure 5.38. In this figure, the first, second, and third rows correspond to Andrews1, Andrews2, and Andrews3 test samples respectively. The first column corresponds to DSM data. The second column corresponds to the ground truth DTM data. The third, fourth, and fifth columns correspond to DTMs generated by the proposed method, SMRF, and gLiDAR respectively.

As can be seen in Figure 5.38, the forest region samples have dense vegetation over a sloped terrain. Some regions also have sharp ridges. For the Andrews1 and Andrews3 test samples, the proposed method lost some detail at some ridge regions. The SMRF method also had the same problem. On the other hand, gLiDAR kept sharp ridges. This will be quantitatively seen in Figure 5.39.

In forest region samples, trees are dense. As a result, almost every point can be labeled as an object even for a small threshold value. In order to compare the object detection performance of the proposed and two other methods, the following strategy is applied. First, the generated DTM is subtracted from DSM and the normalized DSM is obtained. Then, it is thresholded

Table 5.5. Metadata for the NSF Open Topography LiDAR data set.

Name	**Size**	**# points**	**Type**
Andrews1	442 × 488	912,783	dense forest
Andrews2	606 × 710	1,376,563	dense forest
Andrews3	670 × 757	979,133	dense forest
Utah1	521 × 516	1,149,354	residential
Utah2	524 × 763	2,040,449	residential
Utah3	734 × 840	2,885,314	residential
Utah4	693 × 717	1,001,804	residential

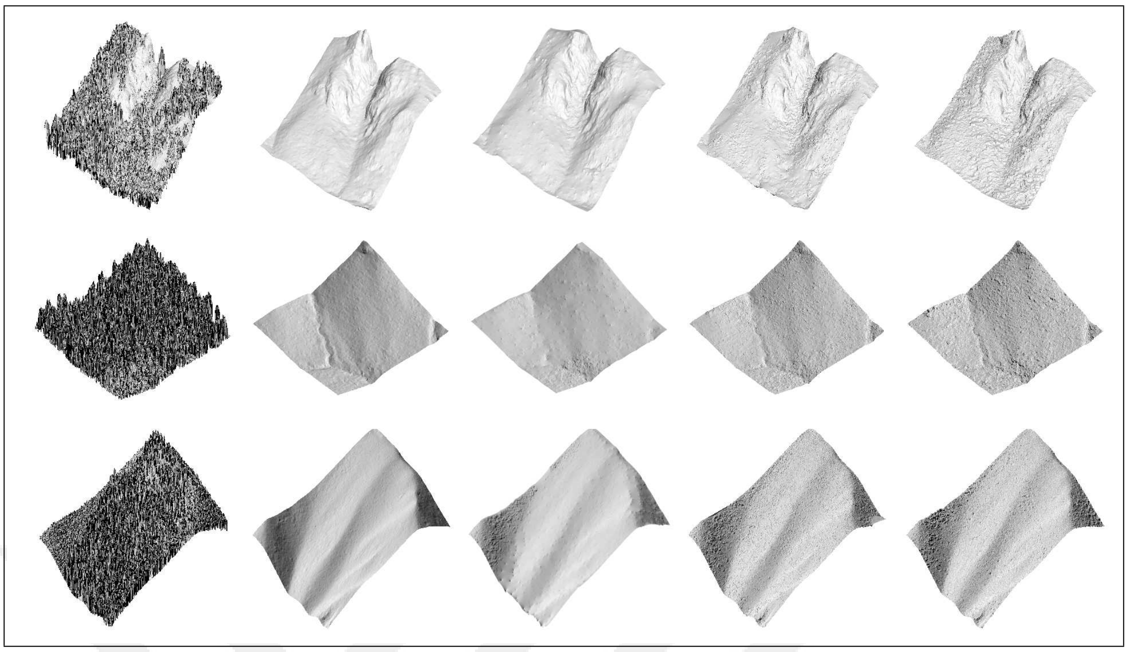


Figure 5.38. DTM generation results on forest region test samples.

with multiple values and Kappa score for each is calculated. This way, object detection results can be obtained for trees having different heights. Otherwise in such a dense forest region, object (mostly tree) detection results may misguide the reader.

Kappa scores for the proposed and other two methods on forest region test samples are provided in Figure 5.39. In this figure, squared, circular, and triangular marks represent the proposed method, SMRF, and gLiDAR respectively. As can be seen in this figure, the obtained Kappa scores are fairly high for all methods. Moreover, for threshold values of 2 m and above, Kappa scores are higher than 80%. Based on these results, it can be claimed that the proposed method performs as well as the other two methods in forest regions.

Next, the proposed method is tested on residential region samples (Utah1, Utah2, Utah3, and Utah4). The Utah1 test site has buildings on a high sloped hillside. Also there is vegetation around buildings. The Utah2 test site has large buildings on a flat terrain. The Utah3 test site contains small buildings on a hillside. The Utah4 test site contains small buildings on a fairly smooth hillside (with low slope). Here, vegetation around buildings is not very dense. As can be seen here, these test sites represent fairly diverse region characteristics. On these samples,

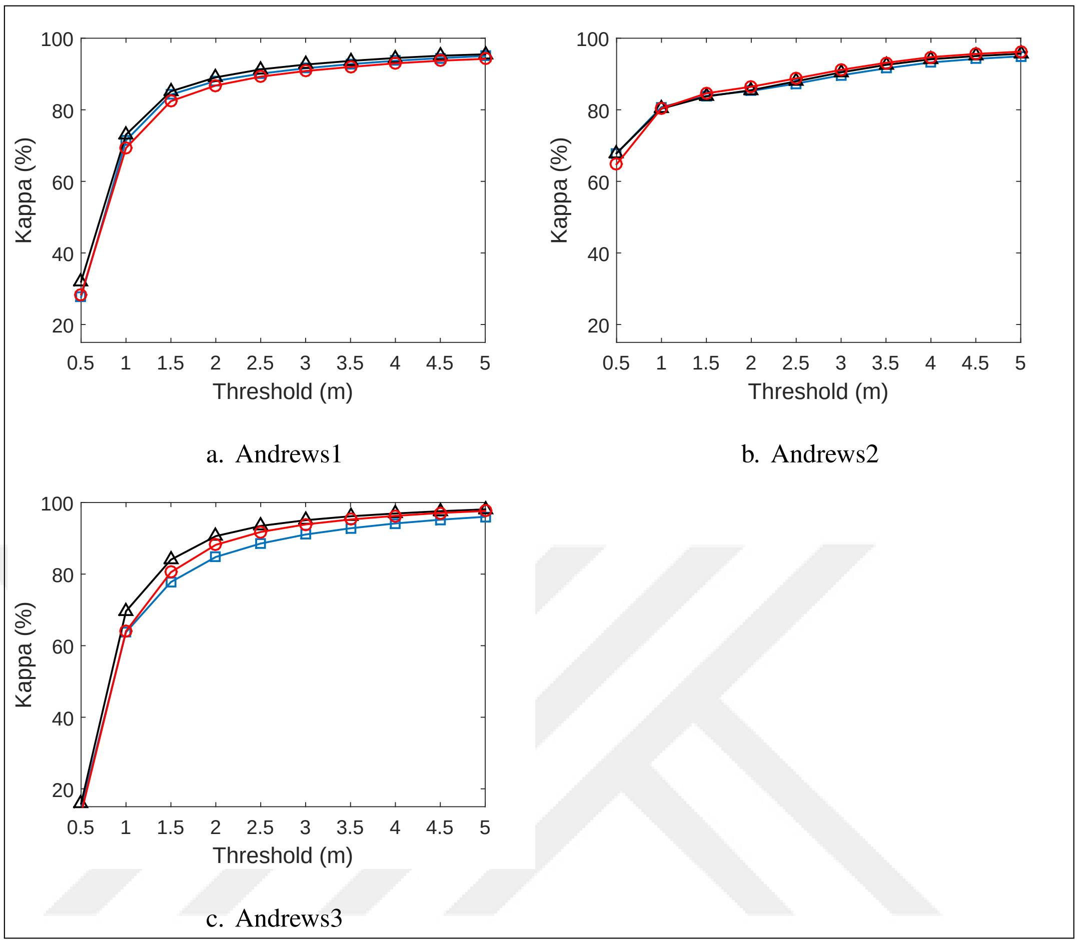


Figure 5.39. Object detection performances in terms of Kappa scores on forest regions.

the results are compared with two different methods. For this test, the maximum window size (wk_{max}) is changed in SMRF to 25 m to obtain a better performance. In a similar manner, in the proposed method the structuring element size, w_i, is set from 1 to 25 m in 10 equally spaced values. The corresponding object detection threshold values are set from 0.1 to 3 m in 10 equally spaced values. Besides, all other parameters are the same as in the forest region data set. The DSM, ground truth DTM, and generated DTMs are provided in Figure 5.40. In this figure, the first four rows correspond to Utah1 to Utah4 test samples respectively. The first column corresponds to DSM data. The second column corresponds to the ground truth DTM data. The third, fourth, and fifth columns correspond to DTMs generated by the proposed method, SMRF, and gLiDAR respectively.

Visual object detection results on residential region test samples is provided in Figure 5.41. In

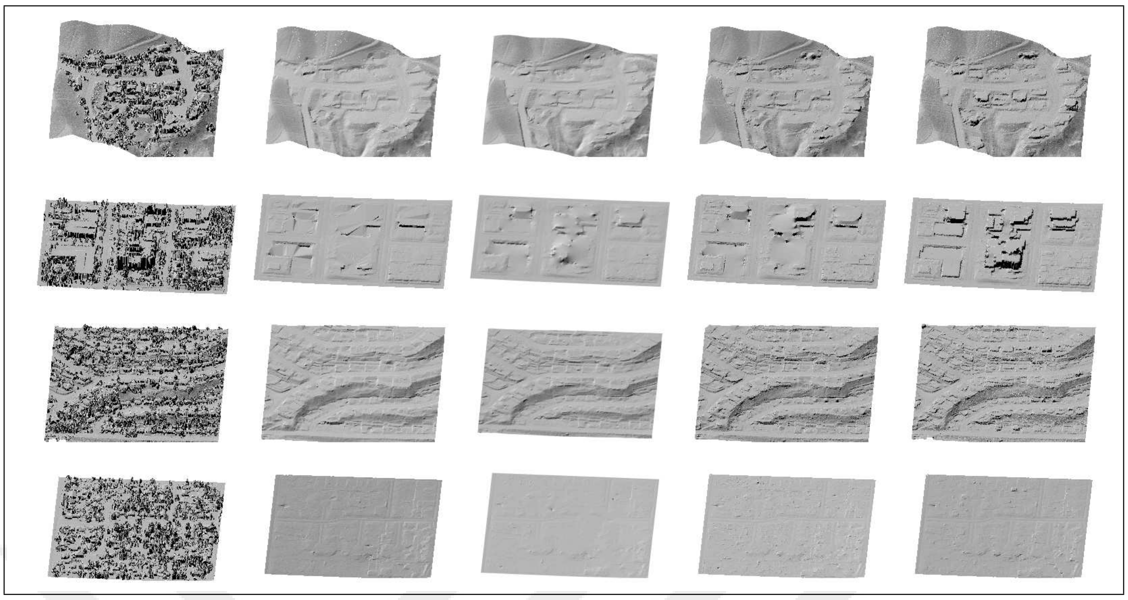


Figure 5.40. DTM generation results on residential region test samples.

this figure, the first four rows correspond to Utah1 to Utah4 test samples respectively. The first column corresponds to the normalized DSM obtained from the ground truth DTM data. The second column corresponds to the thresholded (by one meter) version of this data. The third, fourth, and fifth columns correspond to object detection results obtained by the proposed method, SMRF, and gLiDAR respectively. For the Utah1 sample, there are some false detections at the top right corner of the DSM. Although all buildings are detected correctly, vegetated regions on the sloped parts caused error. On the same test sample, gLiDAR could not detect all parts of the two buildings. SMRF had similar performance as in the proposed method. On the Utah3 test sample, the proposed method was able to detect all buildings with some false alarms towards the edge of buildings and some of the vegetation on the sloped regions. gLiDAR had some problems in detecting the entire layout for some buildings. In the Utah2 test sample, buildings are large. Here all methods had miss detection problems on several buildings. In Utah4, all three methods correctly detected most of the objects.

Kappa scores for the proposed and other two methods on residential region test samples are provided in Figure 5.42. In this figure, squared, circular, and triangular marks represent the proposed method, SMRF, and gLiDAR respectively. As can be seen in this figure, the proposed method has better Kappa scores for all threshold values for the Utah1 and Utah3

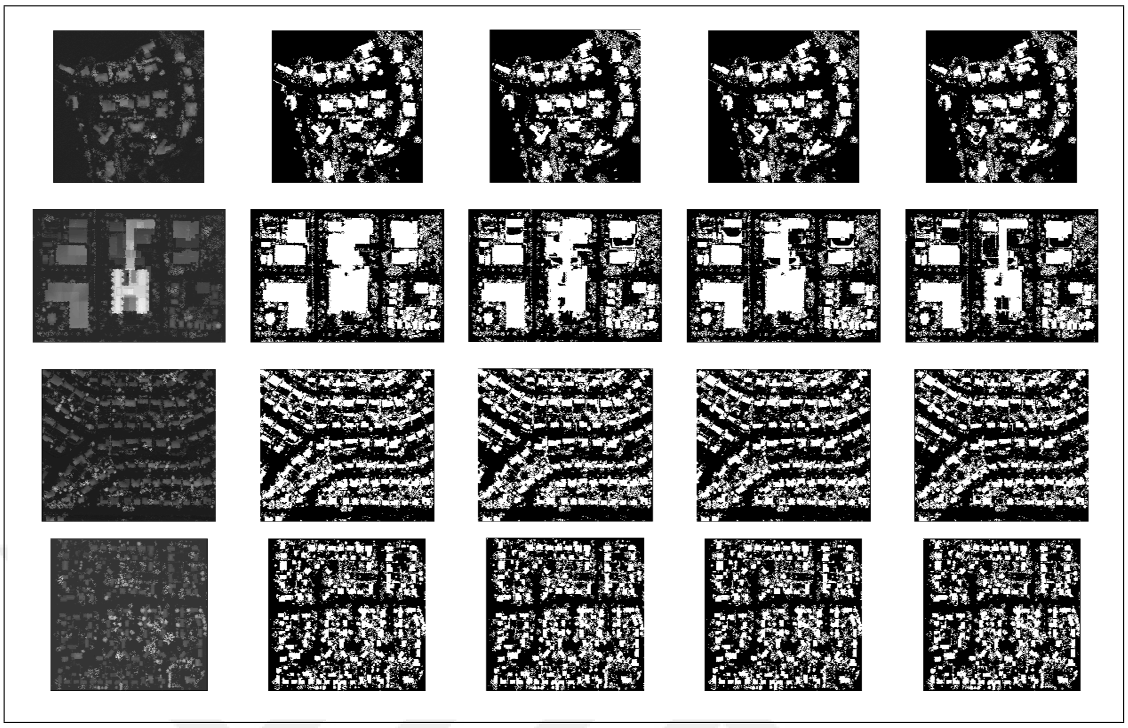


Figure 5.41. Object detection results on Utah test samples.

samples. To note here, these regions have sloped terrain. Therefore, the proposed method can deal with such regions better than the other methods. The remaining Utah2 and Utah4 samples represent regions with flat terrain. In the Utah2 sample, gLiDAR has the worst Kappa score.

Based on the experiments done on the NSF Open Topography data set, some observations can be summarized. First, the proposed method performs as good as the other two methods on forest regions. Second, the proposed method performs fairly well on residential regions compared to the existing two other methods.

The computation time of the proposed and the other two methods are compared in this section. In tests, a PC with Intel Core i7 quad core processor with 8 GB RAM is used. The proposed method is implemented in MATLAB on Windows 7 operating system.

The first comparison is done on the proposed and the other two methods on the ISPRS LiDAR data set composed of 15 samples. The total number of points in this data set is 384,955. The

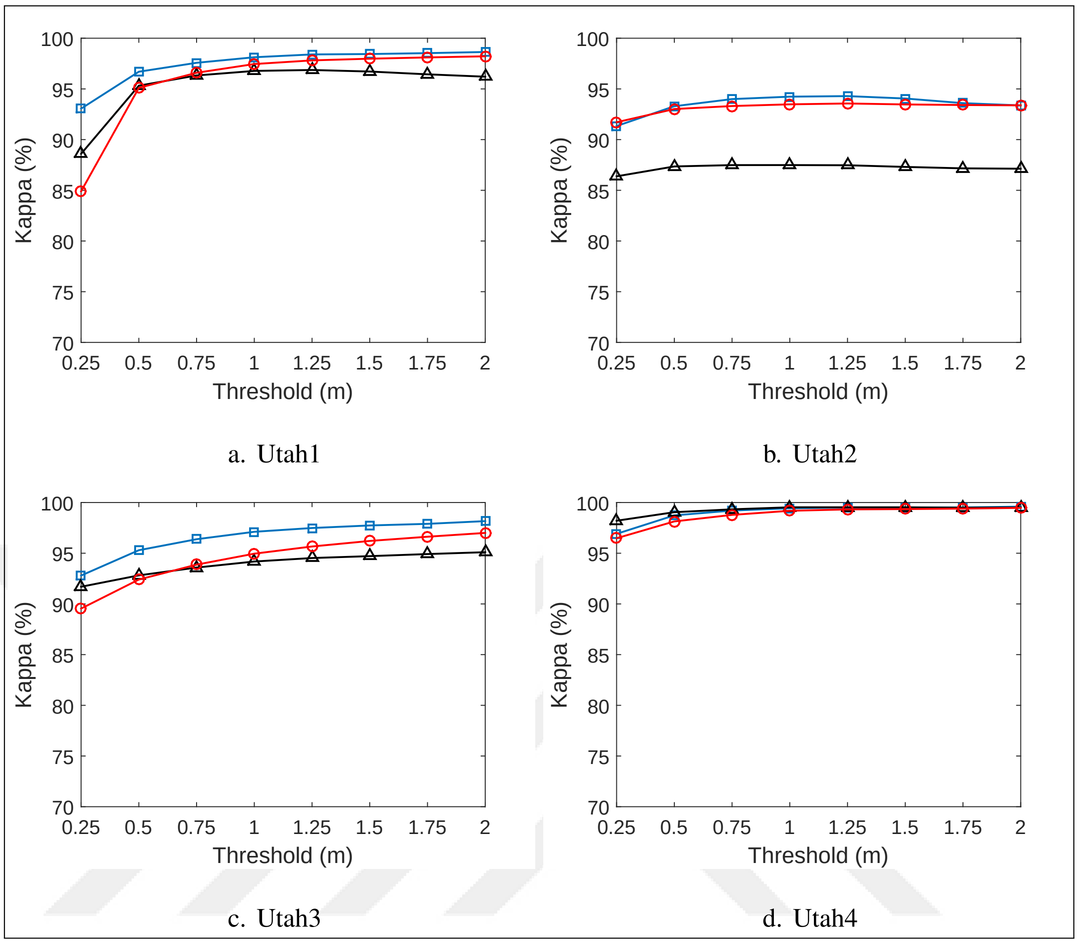


Figure 5.42. Kappa scores of detection results on residential regions.

SMRF method processes all 15 samples in 25.7 seconds. Here, default parameter set is used given in [100]. According to the timing values indicated by Mongus *et al.* [101], it is assumed that gLiDAR needs 7-8 seconds to process all 15 samples. Note that, this method is coded in C++. Therefore, it will be faster than the other two methods coded in MATLAB.

The computation time of the proposed method depends on the interval number. Therefore, the interval number versus computation time is provided in Figure 5.43. As can be seen in this figure, the computation time increases almost linearly with interval number. The interval number is set to 10 in comparing the performance of the proposed method. For this value, the computation time needed to process all 15 samples is 70.3 seconds. Based on this result, it seems that the proposed method is the slowest compared to the other two methods on the ISPRS data set. However, if the user accepts a slightly less performance, then the computation

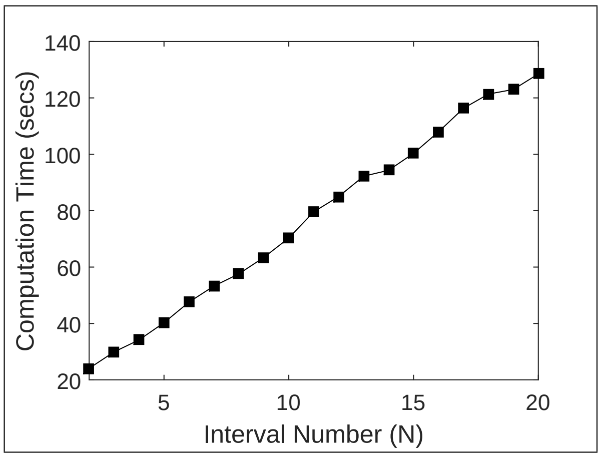


Figure 5.43. Interval number (N) versus the computation time for the ISPRS data set.

time for the proposed method becomes better. One final comment on the computation time of the proposed method is as follows. MATLAB's parallel implementation property is used in obtaining these results. Therefore, the method is run on four processors in a parallel manner. If the number of processors increase, the computation time of the proposed method will decrease since the iterations in the proposed method are independent of each other.

Next, proposed and the other two methods are compared on the NSF Open Topography data set. Here, test sample sizes are large. Moreover, they contain more points to process as can be seen in Table 5.5. The computation times in seconds are tabulated for the proposed and the other two methods in Table 5.6.

As can be seen in Table 5.6, the average computation time for the proposed method is better than gLiDAR and almost the same as SMRF. Again, the interval number for the proposed method is taken as 10 for this timing comparison test. If the interval number decreases, the computation time of the proposed method will become better than the other two methods.

Table 5.6. Computation times in seconds on the NSF Open Topography data set.

Sample	**SMRF [100]**	**gLiDAR [101]**	**Proposed**
Andrews1	24.85	41.40	25.31
Andrews2	42.75	51.94	48.90
Andrews3	41.74	23.33	65.05
Utah1	31.75	60.58	30.44
Utah2	48.55	112.70	50.61
Utah3	68.32	169.85	72.81
Utah4	40.03	20.59	41.94
Average	42.57	68.63	47.86

6. CONCLUSIONS

In this dissertation, novel object detection and shape extraction methods are proposed. Some of the broadly known feature extraction methods are used in a probabilistic voting framework. In this framework, the extracted features are combined to detect the object centers using Bayes theorem. It is shown that using very simple assumptions on object shapes in satellite images, objects such as buildings, trees, ships, cars and airplanes can be detected.

After detecting the object locations, we proposed a shape extraction method on satellite images. Here the modes of object center location pdf are used and a vote back-projection algorithm is proposed. It is shown that the outline of objects are extracted using this algorithm. We extracted the shapes of buildings, tree crowns and ships. We also extracted shapes of objects with multiple parts such as airplanes and cars.

We also proposed two segmentation algorithm for height data. The first method consists of probabilistic voting and morphological segmentation steps. The method is based on the assumption that the object should be higher than its surrounding. However, the proposed method does not depend on DTM generation in extracting objects. Moreover, it is insensitive to the object size to be detected. These two properties are the advantages of the proposed method compared to those available in literature. We tested the proposed method on two different LiDAR and one stereo image based DSM data set. Experimental results show that the proposed method works fairly well in both flat and sloped terrain. Furthermore, it performed better in detecting large buildings compared to the other methods in the literature. There may be some miss detections due to the structure of complex building.

The second method for height data segmentation and filtering is based on two-dimensional EMD. The EMD method is modified by morphology based operations to improve its performance. Moreover, EMD is applied in an iterative manner to obtain the residual signals with different resolution. These serve as adaptive elevation threshold values for crude object detection from DSM data. Then, DTM is generated using this detection result. Afterwards, final objects are detected using DTM and slope based thresholding. The proposed method is compared with the ones in the literature on two publicly available data sets as ISPRS and NSF. For the ISPRS data set, the proposed method performed better than the other methods in the literature in terms of mean Kappa score and the standard deviation of Kappa scores. It also performed better on specific data sets where some other methods failed. For the NSF data set, the proposed method performed as good as the other two methods on seven test samples. On one test sample, it performed better than the other two methods. The proposed method does not require any pre-knowledge of the terrain to work properly. However, if the general characteristics of the terrain is known in advance its performance improves. Two scenarios can be given for this case. For large objects on a flat terrain, the user may increase the maximum structuring element size for a better filtering performance. For small objects on very steep sloped regions, the user may decrease the maximum structuring element size for a better filtering performance. The computation time of the proposed method is comparable with the ones in the literature. Based on these observations, the proposed method can be successfully applied to generate DTM and filter LiDAR data. Wavelets have been used to solve similar problems in image processing as with EMD. If an analogy can be established between EMD and wavelet transform in terms of DTM generation, then new methods may emerge. Besides, researchers can test other modified versions of EMD in connection with the proposed method to improve its performance.

REFERENCES

1. S. Sarkar and K. L. Boyer. Perceptual Organization in Computer Vision: a Review and a Proposal For a Classificatory Structure. *IEEE Transactions on Systems, Man, and Cybernetics*, 23:382-399, 1993.

2. S. Sarkar and K. L. Boyer. A Computational Structure For Preattentive Perceptual Organization: Graphical Enumeration and Voting Methods. *IEEE Transactions on Systems, Man, and Cybernetics*, 24:246-267, 1994.

3. I. Sobel and G. Feldman. A 3x3 Isotropic Gradient Operator For Image Processing. *A Talk at the Stanford Artificial Project*, 271-272, 1968.

4. J. Canny. A Computational Approach To Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 70:679-698, 1986.

5. D. Marr and E. Hildreth. Theory of Edge Detection. *Proceedings of the Royal Society of London B: Biological Sciences*, 207:187-217, 1980.

6. H. P. Moravec. Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover. Technical report, DTIC Document, 1980.

7. C. Harris and M. Stephens. A Combined Corner and Edge Detector. *Proceedings of the 4th Alvey Vision Conference*, 147-151, 1988.

8. E. Rosten and T. Drummond. Fusing Points and Lines For High Performance Tracking. *2005 IEEE 10th International Conference on Computer Vision*, 1508-1515. IEEE, 2005.

9. E. Rosten and T. Drummond. Machine Learning For High-Speed Corner Detection. *Computer vision-ECCV*, 430-443, 2006.

10. E. Rosten, R. Porter, and T. Drummond. Faster and Better: a Machine Learning Approach To Corner Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*,

32:105-119, 2010.

11. D. G. Lowe. Distinctive Image Features From Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60:91-110, 2004.

12. H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded Up Robust Features. *Computer vision-ECCV*, 404-417, 2006.

13. D. H. Ballard. Generalizing the Hough Transform To Detect Arbitrary Shapes. *Pattern Recognition*, 13:111-122, 1981.

14. B. Leibe, A. Leonardis, and B. Schiele. Combined Object Categorization and Segmentation with An Implicit Shape Model. *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

15. J. Gall and V. Lempitsky. Class-Specific Hough Forests For Object Detection. *Decision Forests for Computer Vision and Medical Image Analysis*, 143-157, Springer, 2013.

16. R. B. Irvin and D. M. McKeown. Methods For Exploiting the Relationship Between Buildings and Their Shadows in Aerial Imagery. *IEEE Transactions on Systems, Man, and Cybernetics*, 19:1564-1575, 1989.

17. B. Sirmacek and C. Ünsalan. Building detection from aerial images using invariant color features and shadow information. *Proceedings of ISCIS*, 2008.

18. X. Huang and L. Zhang. Morphological Building/Shadow Index For Building Extraction From High-Resolution Imagery Over Urban Areas. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5:161-172, 2012.

19. A. O. Ok, C. Senaras, and B. Yuksel. Automated Detection of Arbitrarily Shaped Buildings in Complex Environments From Monocular VHR Optical Satellite Imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 51:1701-1717, 2013.

20. F. Rottensteiner, J. Trinder, S. Clode, and K. Kubik. Building Detection by Fusion

of Airborne Laser Scanner Data and Multi-Spectral Images: Performance Evaluation and Sensitivity Analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 62:135-149, 2007.

21. M. Awrangjeb, M. Ravanbakhsh, and C. S. Fraser. Automatic Detection of Residential Buildings Using LIDAR Data and Multispectral Imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65:457-467, 2010.

22. M. Awrangjeb, C. Zhang, and C. S. Fraser. Building Detection in Complex Scenes Thorough Effective Separation of Buildings From Trees. *Photogrammetric Engineering & Remote Sensing*, 78:729-745, 2012.

23. B. Sirmacek and C. Ünsalan. A Probabilistic Framework to Detect Buildings in Aerial and Satellite Images. *IEEE Transactions on Geoscience and Remote Sensing*, 49:211-221, 2011.

24. G. Mountrakis, J. Im, and C. Ogole. Support Vector Machines in Remote Sensing: a Review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66:247-259, 2011.

25. Y. Zhang. Optimisation of Building Detection in Satellite Images by Combining Multispectral Classification and Texture Filtering. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54:50-60, 1999.

26. S. Müller and D. W. Zaum. Robust Building Detection in Aerial Images. *International Archives of Photogrammetry and Remote Sensing*, 36:143-148, 2005.

27. M. Vakalopoulou, K. Karantzalos, N. Komodakis, and N. Paragios. Building Detection in Very High Resolution Multispectral Data with Deep Learning Features. *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 1873-1876. IEEE, 2015.

28. O. Tournaire, M. Bredif, D. Boldo, and M. Durupt. An Efficient Stochastic Approach For Building Footprint Extraction From Digital Elevation Models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65:317-327, 2010.

29. M. Ortner, X. Descombes, and J. Zerubia. Point Processes of Segments and Rectangles For Building Extraction From Digital Elevation Models. *Proceedings of ICASSP 2006*, 2006.

30. A. Brunn and U. Weidner. Extracting Buildings From Digital Surface Models. *International Archives of Photogrammetry and Remote Sensing*, 32:27-34, 1997.

31. B. Sirmacek, H. Taubenböck, P. Reinartz, and M. Ehlers. Performance Evaluation For 3-D City Model Generation of Six Different DSMs From Air and Spaceborne Sensors. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5:59-70, 2012.

32. E. A. S. Galvanin and A. P. D. Poz. Extraction of Building Roof Contours from LiDAR Data Using a Markov Random Field Based Approach. *IEEE Transactions on Geoscience and Remote Sensing*, 50:981-987, 2012.

33. M. Awrangjeb, C. Zhang, and C. S. Fraser. An Improved Building Detection Technique for Complex Scenes. *IEEE International Conference on Multimedia and Expo Workshops*, 2012.

34. M. Larsen. Crown Modelling To Find Tree Top Positions in Aerial Photographs. *3rd International Airborne Remote Sensing Conference and Exhibition*, 1997.

35. L. J. Quackenbush, P. F. Hopkins, and G. J. Kinn. Using Template Correlation To Identify Individual Trees in High Resolution Imagery. *ASPRS Annual Conference*, 2000.

36. M. Wulder, K. O. Niemann, and D. G. Goodenough. Local Maximum Filtering For the Extraction of Tree Locations and Basal Area From High Spatial Resolution Imagery. *Remote Sensing of Environment*, 73:103-114, 2000.

37. D. S. Culvenor. TIDA: An Algorithm For the Delineation of Tree Crowns in High Spatial Resolution Remotely Sensed Imagery. *Elsevier Computers and Geosciences*, 28:33-44, 2002.

38. L. Wang, P. Gong, and G. S. Biging. Individual Tree-Crown Delineation and Treetop Detection in High-Spatial-Resolution Aerial Imagery. *Photogrammetric Engineering and*

Remote Sensing, 70:351-357, 2004.

39. D. Pouliot and D. King. Approaches For Optimal Automated Individual Tree Crown Detection in Regenerating Coniferous Forests. *Canadian Journal of Remote Sensing*, 31:255-267, 2005.

40. N. A. Walsworth and D. J. King. Comparison of Two Tree Apex Delineation Techniques. *Proc. of the International Forum on Automated Interpretation of High Spatial Resolution Digital Imagery for Forestry*, 93-104, 1998.

41. J. Pitkänen. Individual Tree Detection in Digital Aerial Images by Combining Locally Adaptive Binarization and Local Maxima Methods. *Canadian Journal of Forest Research*, 31:832-844, 2001.

42. T. Brandtberg and F. Walter. Automated Delineation of Individual Tree Crowns in High Spatial Resolution Aerial Images by Multiple-Scale Analysis. *Machine Vision and Applications*, 11:64-73, 1998.

43. T. Brandtberg and F. Walter. An Algorithm For Delineation of Individual Tree Crowns in High Spatial Resolution Aerial Images Using Curved Edge Segments At Multiple Scales. *Proceedings of Automated Interpretation of High Spatial Resolution Digital Imagery for Forestry*, 41-54, 1998.

44. D. A. Pouliot, D. J. King, and D. G. Pitt. Development and Evaluation of An Automated Tree Detection Delineation Algorithm For Monitoring Regenerating Coniferous Forests. *Canadian Journal of Forest Research*, 35:2332-2345, 2005.

45. A. N. Skurikhin, S. R. Garrity, N. G. McDowell, and D. M. Cai. Automated Tree Crown Detection and Size Estimation Using Multi-scale Analysis of High-resolution Satellite Imagery. *Remote Sensing Letters*, 4:465-474, 2013.

46. F. Santoro, E. Tarantino, B. Figorito, S. Gualano, and A. M. D'Onghia. A Tree Counting Algorithm for Precision Agriculture Tasks. *International Journal of Digital Earth*, 6:94-102,

2013.

47. F. A. Gougeon. A Crown-Following Approach To the Automatic Delineation of Individual Tree Crowns in High Spatial Resolution Aerial Images. *Canadian Journal of Remote Sensing*, 21:274-284, 1995.

48. F. A. Gougeon. Automatic Individual Tree Crown Delineation Using a Valley-Following Algorithm and Rule-Based System. *International Forum on Automated Interpretation of High Spatial Resolution Digital Imagery for Forestry*, 11-23, 1998.

49. D. G. Leckie, F. A. Gougeon, N. Walsworth, and D. Paradine. Stand Delineation and Composition Estimation Using Semi-Automated Individual Tree Crown Analysis. *Remote Sensing of Environment*, 85:355-369, 2003.

50. F. A. Gougeon and D. G. Leckie. The Individual Tree Crown Approach Applied to Ikonos Images of a Coniferous Plantation Area. *Photogrammetric Engineering and Remote Sensing*, 72:1287-1297, 2006.

51. D. A. Pouliot, D. J. King, F. W. Bell, and D. G. Pitt. Automated Tree Crown Detection and Delineation in High-Resolution Digital Camera Imagery of Coniferous Forest Regeneration. *Remote Sensing of Environment*, 82:322-334, 2002.

52. M. Erikson. Segmentation of Individual Tree Crowns in Colour Aerial Photographs Using Region Growing Supported by Fuzzy Rules. *Canadian Journal of Forest Research*, 33:1557-1563, 2003.

53. P. Bunting and R. Lucas. The Delineation of Tree Crowns in Australian Mixed Species Forests Using Hyperspectral Compact Airborne Spectrographic Imager Data. *Remote Sensing of Environment*, 101:230-248, 2006.

54. F. Rottensteiner, J. Trinder, S. Clode, and K. Kubik. Using the Dempster-Shafer Method For the Fusion of LIDAR Data and Multi-Spectral Images For Building Detection. *Information Fusion*, 6:283-300, 2005.

55. J. Secord and A. Zakhor. Tree Detection in Urban Regions Using Aerial Lidar and Image Data. *IEEE Geoscience and Remote Sensing Letters*, 4:196-200, 2007.

56. J. Reitberger, P. Krzystek, and U. Stilla. Analysis of Full Waveform LIDAR Data For the Classification of Deciduous and Coniferous Trees. *International Journal of Remote Sensing*, 29:1407-1431, 2008.

57. R. Hecht, G. Meinel, and M. F. Buchroithner. Estimation of Urban Green Volume Based On Single-Pulse LiDAR Data. *IEEE Transactions on Geoscience and Remote Sensing*, 46:3832-3840, 2008.

58. A. Ozdarici-Ok. Automatic Detection and Delineation of Citrus Trees from VHR Satellite Imagery. *International Journal of Remote Sensing*, 36:4275-4296, 2015.

59. Y. Ke and L. J. Quackenbush. A Review of Methods for Automatic Individual Tree-crown Detection and Delineation from Passive Remote Sensing. *International Journal of Remote Sensing*, 32:4725-4747, 2011.

60. M. F. Gomes and P. Maillard. *Environmental Applications of Remote Sensing*, Intech, 2016.

61. C. Zhu, H. Zhou, R. Wang, and J. Guo. A Novel Hierarchical Method of Ship Detection from Spaceborne Optical Image Based on Shape and Texture Features. *IEEE Transactions on Geoscience and Remote Sensing*, 48:3446-3456, 2010.

62. N. Proia and V. Page. Characterization of a Bayesian Ship Detection Method in Optical Satellite Images. *IEEE Geoscience and Remote Sensing Letters*, 7:226-230, 2010.

63. C. Corbane, L. Najman, E. Pecoul, L. Demagistri, and M. Petit. A Complete Processing Chain for Ship Detection using Optical Satellite Imagery. *International Journal of Remote Sensing*, 31:5837-5854, 2010.

64. F. Bi, B. Zhu, L. Gao, and M. Bian. A Visual Search Inspired Computational Model for

Ship Detection in Optical Satellite Images. *IEEE Geoscience and Remote Sensing Letters*, 9:749-753, 2012.

65. G. Yang, B. Li, S. Ji, F. Gao, and Q. Xu. Ship Detection From Optical Satellite Images Based on Sea Surface Analysis. *IEEE Geoscience and Remote Sensing Letters*, 11:641-645, 2014.

66. Z. Shi, X. Yu, Z. Jiang, and B. Li. Ship Detection in High-Resolution Optical Imagery Based on Anomaly Detector and Local Shape Feature. *IEEE Transactions on Geoscience and Remote Sensing*, 52:4511-4523, 2014.

67. J. Xu, X. Sun, D. Zhang, and K. Fu. Automatic Detection of Inshore Ships in High-Resolution Remote Sensing Images Using Robust Invariant Generalized Hough Transform. *IEEE Geoscience and Remote Sensing Letters*, 11:2070-2074, 2014.

68. G. Liu, Y. Zhang, X. Zheng, X. Sun, K. Fu, and H. Wang. A New Method on Inshore Ship Detection in High-Resolution Satellite Images Using Shape and Context Information. *IEEE Geoscience and Remote Sensing Letters*, 11:617-621, 2014.

69. J. Tang, C. Deng, G. B. Huang, and B. Zhao. Compressed-Domain Ship Detection on Spaceborne Optical Image Using Deep Neural Network and Extreme Learning Machine. *IEEE Transactions on Geoscience and Remote Sensing*, 53:1174-1185, 2015.

70. W. Yao, S. Hinz, and U. Stilla. Automatic Vehicle Extraction From Airborne LiDAR Data of Urban Areas Aided by Geodesic Morphology. *Pattern Recognition Letters*, 31:1100-1108, 2010.

71. T. Moranduzzo and F. Melgani. Automatic Car Counting Method for Unmanned Aerial Vehicle Images. *IEEE Transactions on Geoscience and Remote Sensing*, 52:1635-1647, 2014.

72. D. A. Grejner-Brzezinska, C. K. Toth, and E. Paska. *Airborne Remote Sensing Supporting Traffic Flow Estimation*, Taylor and Francis, London, 2007.

73. F. Meyer, S. Hinz, A. Laika, D. Weihing, and R. Bamler. Performance Analysis of the TerraSAR-X Traffic Monitoring Concept. *ISPRS Journal of Photogrammetry and Remote Sensing*, 61:225-242, 2006.

74. H. Runge, S. Suchandt, A. Kotenkov, H. Breit, M. Vonavka, and U. Balss. Space Borne SAR Traffic Monitoring. *Proceedings on International Radar Symposium*, 2007.

75. W. Yao, S. Hinz, and U. Stilla. Traffic Monitoring From Airborne LIDAR-Feasibility, Simulation and Analysis. *Proceedings of XXI Congress on IAPRSSGS*, 2008.

76. X. Jin and C. H. Davis. Vector Guided Vehicle Detection From High-Resolution Satellite Imagery. *Proceedings of IGARSS'04*, 1095-1098, 2004.

77. J. Leitloff, S. Hinz, and U. Stilla. Vehicle Detection in Very High Resolution Satellite Images of City Areas. *IEEE Transactions on Geoscience and Remote Sensing*, 48:2795-2806, 2010.

78. G. Cheng, J. Han, L. Guo, X. Qian, P. Zhou, X. Yao, and X. Hu. Object Detection in Remote Sensing Imagery Using a Discriminatively Trained Mixture Model. *ISPRS Journal of Photogrammetry and Remote Sensing*, 85:32-43, 2013.

79. J. Han, P. Zhou, D. Zhang, G. Cheng, L. Guo, Z. Liu, S. Bu, and J. Wu. Efficient, Simultaneous Detection of Multi-Class Geospatial Targets Based On Visual Saliency Modeling and Discriminative Learning of Sparse Coding. *ISPRS Journal of Photogrammetry and Remote Sensing*, 89:37-48, 2014.

80. Z. Lei, T. Fang, H. Huo, and D. Li. Rotation-Invariant Object Detection of Remotely Sensed Images Based On Texton Forest and Hough Voting. *IEEE Transactions on Geoscience and Remote Sensing*, 50:1206-1217, 2012.

81. Y. Yu, H. Guan, D. Zai, and Z. Ji. Rotation-And-Scale-Invariant Airplane Detection in High-Resolution Satellite Images Based On Deep-Hough-Forests. *ISPRS Journal of Photogrammetry and Remote Sensing*, 112:50-64, 2016.

82. S. Qiu, G. Wen, and Y. Fan. Occluded Object Detection in High-Resolution Remote Sensing Images Using Partial Configuration Object Model. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2017.

83. G. Sithole and G. Vosselman. Experimental Comparison of Filter Algorithms For Bare-Earth Extraction From Airborne Laser Scanning Point Clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 59:85-101, 2004.

84. G. Vosselman. Slope Based Filtering of Laser Altimetry Data. *International Archives of Photogrammetry and Remote Sensing*, 935-942, 2000.

85. G. Sithole. Filtering of Laser Altimetry Data Using a Slope Adaptive Filter. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 203-210, 2001.

86. J. Shan and S. Aparajithan. Urban DEM Generation From Raw LIDAR Data: a Labeling Algorithm and Its Performance. *Photogrammetric Engineering and Remote Sensing*, 71:217-226, 2005.

87. K. Kraus and N. Pfeifer. A New Method For Surface Reconstruction From Laser Scanner Data. *International Archives of Photogrammetry and Remote Sensing*, 80-86, 1997.

88. V. T. Thuy, R. Yokoyama, F. Yamazaki, and M. Tokunaga. Wavelet-Based System For Classification of Airborne Laser Scanner Data. *Proceedings of IEEE International Geoscience and Remote Sensing Symposium*, 4404-4406, 2003.

89. H. S Lee and N. H. Younan. DTM Extraction of LiDAR Returns Via Adaptive Processing. *IEEE Transactions on Geoscience and Remote Sensing*, 41:2063-2069, 2003.

90. M. A. Brovelli, M. Cannata, and U. M. Longoni. LIDAR Data Filtering and DTM Interpolation Within GRASS. *Transactions in GIS*, 8:155-174, 2004.

91. K. Zhang and D. Whitman. Comparison of Three Algorithms For Filtering Airborne

Lidar Data. *Photogrammetric Engineering and Remote Sensing*, 71:313-324, 2005.

92. D. Mongus and B. Zalik. Parameter-Free Ground Filtering of LiDAR Data For Automatic DTM Generation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 67:1-12, 2012.

93. C. Chen, Y. Li, W. Li, and H. Dai. A Multiresolution Hierarchical Classification Algorithm For Filtering Airborne LiDAR Data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 82:1-9, 2013.

94. J. Zhang and X. Lin. Filtering Airborne LiDAR Data by Embedding Smoothness-Constrained Segmentation in Progressive TIN Densification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 81:44-59, 2013.

95. D. Mongus and B. Zalik. Computationally Efficient Method For the Generation of a Digital Terrain Model From Airborne LiDAR Data Using Connected Operators. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7:340-351, 2014.

96. H. Hu, Y. Ding, Q. Zhu, B. Wu, H. Lin, Z. Du, Y. Zhang, and Y. Zhang. An Adaptive Surface Filter For Airborne Laser Scanning Point Clouds by Means of Regularization and Bending Energy. *ISPRS Journal of Photogrammetry and Remote Sensing*, 92:98-111, 2014.

97. Z. Ding, Y. Yu, B. Wang, and L. Zhang. An Approach for Visual Attention Based On Biquaternion and Its Application for Ship Detection in Multispectral Imagery. *Neurocomputing*, 76:9-17, 2012.

98. K. Zhang, J. Yan, and S. C. Chen. Automatic Construction of Building Footprints From Airborne LIDAR Data. *IEEE Transactions on Geoscience and Remote Sensing*, 44:2523-2533, 2006.

99. Q. Chen, P. Gong, D. Baldocchi, and G. Xie. Filtering Airborne Laser Scanning Data with Morphological Methods. *Photogrammetric Engineering and Remote Sensing*, 73:175-185, 2007.

100. T. J. Pingel, K. C. Clarke, and W. A. McBride. An Improved Simple Morphological Filter For the Terrain Classification of Airborne LIDAR Data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 77:21-30, 2013.

101. D. Mongus, N. Lukac, and B. Zalik. Ground and Building Extraction From LiDAR Data Based On Differential Morphological Profiles and Locally Fitted Surfaces. *ISPRS Journal of Photogrammetry and Remote Sensing*, 93:145-156, 2014.

102. S. Filin. Surface Clustering From Airborne Laser Scanning Data. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 34:119-124, 2002.

103. S. Filin and N. Pfeifer. Segmentation of Airborne Laser Scanning Data Using a Slope Adaptive Neighborhood. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60:71-80, 2006.

104. T. Rabbani, F. Van den Heuvel, and G. Vosselmann. Segmentation of Point Clouds Using Smoothness Constraint. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36:248-253, 2006.

105. M. Bartels and H. Wei. Segmentation of LiDAR Data Using Measures of Distribution. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36:426-31, 2006.

106. B. Yunfei, L. Guoping, C. Chunxiang, L. Xiaowen, Z. Hao, H. Qisheng, B. Linyan, and C. Chaoyi. Classification of LiDAR Point Cloud and Generation of DTM From LiDAR Height and Intensity Data in Forested Area. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37:313-318, 2008.

107. M. Bartels and H. Wei. Threshold-Free Object and Ground Point Separation in LiDAR Data. *Pattern Recognition Letters*, 31:1089-1099, 2010.

108. F. Crosilla, D. Macorig, M. Scaioni, I. Sebastianutti, and D. Visintini. LiDAR Data Filtering and Classification by Skewness and Kurtosis Iterative Analysis of Multiple Point

Cloud Data Categories. *Applied Geomatics*, 5:225-240, 2013.

109. A. B. Jahromi, M. J. V. Zoej, A. Mohammadzadeh, and S. Sadeghian. A Novel Filtering Algorithm for Bare-Earth Extraction From Airborne Laser Scanning Data Using an Artificial Neural Network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 4:836-843, 2011.

110. W. T. Freeman and E. H. Adelson. The Design and Use of Steerable Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:891-906, 1991.

111. J. F. Lalonde, A. A. Efros, and S. G. Narasimhan. Estimating Natural Illumination From a Single Outdoor Image. *2009 IEEE 12th International Conference on Computer Vision*, 183-190, IEEE, 2009.

112. D. L. Waltz. Generating Semantic Descriptions From Drawings of Scenes with Shadows, Technical report, MIT, 1972.

113. N. Otsu. A Threshold Selection Method from Gray-level Histograms. *IEEE Transactions on System, Man, and Cybernetics*, 9:62-66, 1979.

114. D. Cohen. *Precalculus: With Unit Circle Trigonometry*, Cengage Learning, 2005.

115. V. N. Vapnik and V. Vapnik. *Statistical Learning Theory 1*, Wiley New York, 1998.

116. J. Davis and M. Goadrich. The Relationship Between Precision-Recall and ROC Curves. *Proceedings of the 23rd International Conference on Machine Learning*, 233-240, 2006.

117. ISPRS Test on Extracting DEMs From Point Clouds, http://www.itc.nl/isprswgiii-3/filtertest [retrieved 21 July 2015].

118. C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The Quickhull Algorithm For Convex Hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22:469-483, 1996.

119. T. F. Chan and L. A. Vese. Active Contours Without Edges. *IEEE Transactions on*

Image Processing, 10:266-277, 2001.

120. J. Gong, Q. Li, H. S. Zhu, and Y. Zhou. Effects of Various Factors on the Accuracy of DEMs: An Intensive Experimental Investigation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 1113-1117, 2000.

121. B. Sirmacek, H. Taubenbock, P. Reinartz, and M. Ehlers. Performance Evaluation for 3-D City Model Generation of Six Different DSMs From Air- and Spaceborne Sensors. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5:59-70, 2012.

122. R. Dinuls, G. Erins, A. Lorencs, I. Mednieks, and J. Sinica-Sinavskis. Tree Species Identification in Mixed Baltic Forest Using LiDAR and Multispectral Data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5:594-603, 2012.

123. A. O. Onojeghuo and G. A. Blackburn. Characterising Reedbeds Using LiDAR Data: Potential and Limitations. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6:935-941, 2013.

124. Open Topography: High Resolution Topography Data and Tools, http://opentopo.sdsc.edu/gridsphere/gridsphere?cid=geonlidar [retrieved 21 July 2015].

125. Rafael C Gonzalez and Richard E Woods. *Digital Image Processing*, Prentice Hall, 2002.

126. M. M. Mano and M. D. Ciletti. *Digital Design*, Pearson, 2007.

127. A.H. Özcan and C. Ünsalan. LiDAR Data Filtering and DTM Generation Using Empirical Mode Decomposition. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10:360-371, 2017.

128. M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image Inpainting. *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, 417-424, 2000.

129. J. Dericco. File Exchange MATLAB Central, http://www.mathworks.com/matlabcentral/fileexchange/4551-inpaint-nans [retrieved 21 July 2015].

130. J. Cohen. Weighted Kappa: Nominal Scale Agreement Provision For Scaled Disagreement Or Partial Credit. *Psychological Bulletin*, 70:213-220, 1968.

131. M. Rutzinger, F. Rottensteiner, and N. Pfeifer. A Comparison of Evaluation Techniques For Building Extraction From Airborne Laser Scanning. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2:11-20, 2009.

132. T. J. Pingel. A Simple Morphological Filter for Fround Identification of LIDAR Data, http://tpingel.org/code/smrf/smrf.html [retrieved 21 July 2015].

133. GeMMALab gLIDAR Software Homepage, http://gemma.uni-mb.si/gLiDAR/about.html [retrieved 21 July 2015].

134. P. dAngelo and P. Reinartz. Semiglobal Matching Results On the ISPRS Stereo Matching Benchmark. *ISPRS Hannover Workshop High-Resolution Earth Imaging for Geospatial Information*, 2011.

135. 2D Semantic Labeling Vaihingen, http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-vaihingen.html [retrieved 21 December 2015].

136. B. Demir and S. Erturk. Empirical Mode Decomposition of Hyperspectral Images for Support Vector Machine Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 48:4071-4084, 2010.

137. B. Demir, S. Erturk, and M. K. Gullu. Hyperspectral Image Classification Using Denoising of Intrinsic Mode Functions. *IEEE Geoscience and Remote Sensing Letters*, 8:220-224, 2011.

138. E. T. Gormus, N. Canagarajah, and A. Achim. Dimensionality Reduction of Hyperspectral Images Using Empirical Mode Decompositions and Wavelets. *IEEE Journal*

of Selected Topics in Applied Earth Observations and Remote Sensing, 5:1821-1830, 2012.

139. A. Erturk, M. K. Gullu, and S. Erturk. Hyperspectral Image Classification Using Empirical Mode Decomposition With Spectral Gradient Enhancement. *IEEE Transactions on Geoscience and Remote Sensing*, 51:2787-2798, 2013.

140. Z. He, Q. Wang, Y. Shen, and M. Sun. Kernel Sparse Multitask Learning for Hyperspectral Image Classification With Empirical Mode Decomposition and Morphological Wavelet-Based Features. *IEEE Transactions on Geoscience and Remote Sensing*, 52:5150-5163, 2014.

141. N. E. Huang, Z. Shen, S. R. Long, M. L. Wu, H. H. Shih, Q. Zheng, N. C. Yen, C. C. Tung, and H. H. Liu. The Empirical Mode Decomposition and Hilbert Spectrum for Non-linear and Non-stationary Time Series Analysis. *Procedings of the Royal Society*, 903-995, 1998.

142. J. C. Nunes, Y. Bouaoune, E. Delechelle, O. Niang, and P. Bunel. Image Analysis by Bidimensional Empirical Mode Decomposition. *Image and Vision Computing*, 21:1019-1026, 2003.

143. J. C. Nunes and E. Deléchelle. Empirical Mode Decomposition: Applications On Signal and Image Processing. *Advances in Adaptive Data Analysis*, 1:125-175, 2009.

144. M. S. Koh, E. Rodriguez-Marek, and T. R. Fischer. A New Two Dimensional Empirical Mode Decomposition For Images Using Inpainting. *IEEE 10th International Conference on Signal Processing*, 13-16, 2010.

145. Y. Q. Sun, M. S. Koh, E. Rodriguez-Marek, and C. Talarico. A New Infrared Image Fusion Method Using Empirical Mode Decomposition and Inpainting. *IEEE 18th International Conference on Image Processing*, 1477-1480, 2011.

146. A. H. Özcan. Yeditepe University Computer Vision Research Laboratory Homepage, http://vision.yeditepe.edu.tr/DTMcodes.rar [retrieved 21 September 2015].

147. N. Pfeifer, T. Reiter, C. Briese, and W. Rieger. Interpolation of High Quality Ground Models From Laser Scanner Data in Forested Areas. *International Archives of Photogrammetry and Remote Sensing*, 31-36, 1999.

148. P. Axelsson. DEM Generation From Laser Scanner Data Using Adaptive TIN Models. *International Archives of Photogrammetry and Remote Sensing*, 111-118, 2000.

149. X. Meng, L. Wang, J. L. Silvan-Cardenas, and N. Currit. A Multi-Directional Ground Filtering Algorithm For Airborne LIDAR. *ISPRS Journal of Photogrammetry and Remote Sensing*, 64:117-124, 2009.

150. M. Isenburg, Y. Liu, J. Shewchuk, J. Snoeyink, and T. Thirion. LAStools: Converting, Filtering, Viewing, Processing, and Compressing LIDAR Data in LAS Format, http://www.cs.unc.edu/ isenburg/tin2dem/ [retrieved 21 July 2015].