OBJECT DETECTION IN AERIAL AND SATELLITE IMAGES

by

Beril Sırmaçek

Submitted to the Institute of Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy
in
Electrical and Electronics Engineering

Yeditepe University

2009

OBJECT DETECTION IN AERIAL AND SATELLITE IMAGES

APPROVED BY:

Assoc. Prof. Dr. Cem Ünsalan (Thesis Supervisor)	. . . . .
Prof. Dr. Aytül Erçil	. . . . .
Assist. Prof. Dr. Dionysis Goularas	. . . . .
Assist. Prof. Dr. Duygun Erol Barkana	. . . . .
Assist. Prof. Dr. Selim Aksoy	. . . . .

DATE OF APPROVAL: .....

ACKNOWLEDGEMENTS

I think the last two years of my life constitute the most important part in my academic studies. Each step of this thesis work was very challenging and very interesting at the same time. Through working on this interesting topic, I have been able to meet and collaborate with a large group of excellent scientist.

I am grateful to my supervisor Cem Ünsalan for suggesting me this exciting research field. Without his guidance and even contributions down to the details of mathematical derivations, I would not have been able to complete the more complex parts of this work. I also know that no thesis can be finalized without being approved by the thesis committee. I would like to thank to Duygun Erol Barkana, and Dionysis Goularas for fruitful discussions and comments. I am grateful to Aytül Erçil for guiding me. I also would like to thank to Deniz Pazarcı and Korkut Yeğin for giving their times to listen to me whenever I could not sooth my feelings.

I owe special thanks to my parents, all relatives, friends and colleagues in our department for believing in me. My brother Berk Sırmaçek, Yeşim Yazgan, Tuğba Haykır, my dear grandmother and all of the other people gave me positive influence to complete this thesis work. Finally, I am grateful to my dear husband İlker Oran for his endless support. I especially thank him for his great patience, and helps even for generating my groundtruth images.

ABSTRACT

OBJECT DETECTION IN AERIAL AND SATELLITE IMAGES

Very high resolution satellite and aerial images provide valuable information to researchers. With their availability, there has been much interest to extract man-made objects from such imageries. Among these, detection of objects such as buildings, road segments, and urban area boundaries play crucial roles especially for municipalities, government agencies, rescue teams, military, and other civil agencies. For a human expert, manually extracting this valuable information is tedious and prone to errors. One possible solution to extract this information is developing automated techniques. Unfortunately, the solution is not straightforward if standard image processing and pattern recognition techniques are used.

In this thesis, we propose new approaches using several local and semi-local invariant features (such as SIFT, Gabor features, gradient features, and color invariants) to automatically detect man-made objects in remotely sensed images. These invariant features are very powerful in detecting objects under various imaging conditions (like illumination, viewing angle, etc). However, extraction of invariant features is not sufficient for detecting objects. Therefore, we further formalize the problem by developing graph theoretical, probabilistic, and region based methods, to extract structural information to verify object appearance. Using these mathematical techniques, we first develop methods to detect urban area boundaries. We also formulate some measures to estimate the degree of urbanization in detected urban areas. Having detected urban area boundaries, we develop our algorithms to detect separate buildings and road segments in these regions. Besides, we also developed novel methods to find approximate building shapes and to estimate damaged structures in color images. To detect road segments, we proposed two methods using local features and color information separately. We tested the robustness of our algorithms using a diverse data set including very high resolution panchromatic Ikonos satellite and aerial images. Experimental results indicate the high performance and usefulness of our object detection approaches on such a diverse image dataset.

ÖZET

HAVA VE UYDU GÖRÜNTÜLERİNDE NESNE TANIMA

Çok yüksek çözünürlüklü uydu görüntüleri ve hava fotoğrafları araştırmacılara önemli bilgiler sunmaktadır. Bunların ticari olarak bulunabilir olması ile birlikte, bu görüntülerden insan yapımı nesnelerin çıkarımına ilgi oldukça artmıştır. Bu nesnelerin arasında binaların, yolların ve yerleşim bölgelerinin sınırlarının bulunması özellikle belediyeler, bazı bakanlıklar, arama-kurtarma ekipleri, askeri birlikler ve diğer sivil kuruluşlar için önemli rol oynamaktadır. Bu önemli bilgilerin uzman bir kişi tarafından çıkarılması çok uğraştırıcı ve hatalara açıktır. Ne yazık ki, alışılagelmiş görüntü işleme ve örüntü tanıma teknikleri ile bu probleme çözüm bulmak mümkün değildir.

Bu çalışmada, uzaktan algılanan görüntülerde insan yapımı nesneleri tanıma amacı ile yerel ve yarı-yerel özniteliklerden (SIFT, Gabor, gradyan ve renk değişmezleri gibi) yararlanarak yeni yöntemler önerdik. Bu değişmez öznitelikler faklı görüntüleme şartları altında (aydınlanma, bakış açısı gibi) nesne tanıma için oldukça güçlüdürler. Ancak, değişmez özniteliklerin çıkarımı nesne tanımak ve yerini tespit etmek için yeterli değildir. Bu nedenle, yapısal özellikleri ortaya çıkarmak ve nesneyi tespit edebilmek için probleme graf teorisi, olasılıksal yöntemler ve alan temelli yöntemlerden faydalanarak çözüm önerdik. Bu matematiksel tekniklere dayanarak, öncelikle yerleşim bölgelerinin sınırlarını tespit ettik. Bulunan yerleşim biriminin kentleşme derecesini tespit edebilmek için bazı ölçütler önerdik. Yerleşim bölgelerinin sınırlarını bulduktan sonra ayrık binaları ve yolları tespit edebilmek için yöntemler geliştirdik. Ayrık binaları tespit ettikten sonra bina şekillerini ve hasarlı binaları otomatik tespit edebilmek için yeni yöntemler önerdik. Yolları bulmak için de yerel öznitelikleri ve renk bilgisini kullanan yöntemler önerdik. Algoritmalarımızın dayanıklılığını test edebilmek için birbirinden oldukça farklı çok yüksek çözünürlüklü uydu görüntüleri ve hava fotoğraflarından oluşan veri seti kullandık. Deneysel sonuçlar, önerdiğimiz nesne tanıma yöntemlerinin çeşitli görüntülerden oluşan veri seti üzerinde yüksek başarım ile çalıştığını ve yöntemlerin pratik kullanılabilirliğini göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS/ABBREVIATIONS

B	Blue Color Band of Image
$C_{k,l}(x, y)$	Detected road centers
$c_t h$	User controlled threshold to stop road tracking
$D(x, y)$	Differences of two smoothed images
D_θ^k	Dilated kth steerable filter feature in θ direction
e	Error in detection or ordering
E	Graph edges
f	Descriptor vector for SIFT local feature
F_θ^k	kth steerable filter feature in θ direction
G	Green Color Band of Image
h	Window width in Harris corner detection
$I_b(x, y)$	Bilateral filtered test image
$I_g(x, y)$	Grayscale test image
$I_x(x, y)$	Gradients of test image in x direction
$I_y(x, y)$	Gradients of test image in y direction
$J_\theta(x, y)$	Steerable filter response in θ filtering direction
k_f	Descriptor vector for Gabor local feature
k_g	Descriptor vector for GMSR local feature
k_h	Descriptor vector for harris local feature
$K(x, y)$	Kernel function
$L(x, y)$	Smoothed image with Gaussian low-pass filter
m_1	Number of voting local features
m_2	Normalized sum of votes
m_3	Maximum vote
m_4	Normalized urban area
m_5	Normalized sum of votes in urban region
$p_b(x, y)$	Estimated pdf in building detection
P_d	Percentage of True Detections
P_f	Percentage of False Alarms
$p_g(x, y)$	Estimated pdf using Gabor features
$p_g(x, y)$	Estimated pdf using gradient magnitude support region features

$p_h(x, y)$	Estimated pdf using Harris corner features
R	Red Color Band of Image
s_f	Sort of the image in sequence by feature
s_o	Ordinal sort of the image in sequence
V	Graph vertices
$V(x, y)$	Voting matrix
w	Weight of local feature
$w_{k,l}$	Estimated road width
μ	Mean of road pixel responses
σ	Smoothing parameter of Gaussian function
τ_g	Smoothing parameter in Harris corner detection
ψ_r	Illumination Invariant Red Color Index
ψ_b	Illumination Invariant Blue Color Index
BF	Bilateral Filter
FA	False Alarm
$GMSR$	Gradient Magnitude Support Regions
GVF	Gradient Vector Flow
HCV	Hue Chroma Value
HSI	Hue Saturation Intensity
HSV	Hue Saturation Value
$NDVI$	Normalized Difference Vegetation Index
PCA	Principle Component Analysis
pdf	probability density function
$SIFT$	Scale Invariant Feature Transform
TP	True Positive
UA	Urban Area
VHR	Very High Resolution
YC_bC_r	Luma Blue-chroma Red-chroma
YIQ	Luma In-phase Quadrature

1. INTRODUCTION

Remote sensing is the science of obtaining and interpreting information from a distance, using special sensors that are not in physical contact with the object being observed. Remote Sensing is formally defined by the American Society for Photogrammetry and Remote Sensing (ASPRS) as follows; "Photogrammetry and remote sensing are the art, science, and technology of obtaining reliable information about physical objects and the environment, through the process of recording, measuring and interpreting imagery and digital representations of energy patterns derived from non-contact sensor systems" [1]. In this thesis, saying 'remote sensing' or 'remotely sensed image' we consider acquisition of information (image) via aerial or satellite sensors.

Satellite images provide information (in terms of radar, infrared, and multispectral imagery) obtained from sensors on geostationary or orbiting satellites. If the same information is obtained from sensors aboard manned or unmanned aircraft, then it is called an aerial image. The resolution of earlier satellite images (such as Landsat with almost $15\ m$ spatial resolution) would not allow detecting separate man-made or natural objects. Therefore, researchers mostly focused on extracting the region properties and urban area boundaries from these images.

As the advent of very high resolution (VHR) satellite imagery (such as Ikonos and Quikbird), it became possible to observe separate man-made objects. Besides region properties, extracting man-made objects in very high resolution satellite images may help researchers in various ways, such as automated mapping. Although the resolution of images has reached an acceptable level, unfortunately it is still tedious for a human expert to manually extract man-made objects in a given remotely sensed image. One main reason is the total number of objects in the scene. The other reason is the area of the region that these images cover. To solve these problem, researchers focus on developing automated techniques to detect separate man-made objects from remotely sensed images.

Among man-made objects, buildings play an important role. Therefore, robust detection of buildings in very high resolution satellite and aerial images requires a specific consideration. Detecting buildings in satellite images is still a difficult task for several reasons. Buildings may be imaged from different view points. Illumination and contrast in the image may not be sufficient for detecting them. There may be several other structures, such as nearby trees and street seg-

ments making the building detection problem harder. In addition to these difficulties, buildings do not have standard size and shape. All these issues make building detection a hard problem to solve with one generic algorithm.

Road segments are also very important man-made objects in an urban scene. Therefore, detecting road segments from remotely sensed images became a popular research topic in recent years. Especially city planners need automated road detection systems to update previously constructed road maps. Although road shapes do not vary as buildings, sometimes it can be more difficult to detect them. Roads are generally occluded by other objects like buildings and trees. To solve this object detection problem, more intelligent computer vision techniques are needed and researchers studied on developing automated detection methods using very high resolution satellite and aerial images. Ünsalan [2] provided a detailed literature review on both building and road segment extraction methods.

In this thesis, we developed novel approaches to detect man-made objects in Ikonos satellite and aerial images. The Ikonos satellite has a sensor to capture four multispectral bands with $4\ m$ spatial resolution and the panchromatic band with $1\ m$ spatial resolution. The color aerial image set has a $0.3\ m$ spatial resolution. We considered detecting urban area boundaries, separate buildings, road segments in these images. To do so, we used local or semi-local invariant features.

There are a number of local invariant features that have been proposed for various visual recognition tasks. Using these features, robust object detection can be performed if the object appears under different conditions (such as illumination, viewing angle, different scales, etc.). Local invariant features are generally preferred when the appearance of the objects are not controlled. Since object appearance is also not controlled in our problem, we benefit from local and semi-local invariant features to generate robust object detection systems. In addition to using well-known local invariant features, we also propose new features that fit our problem.

Using local features, we formalize our object detection problem by developing graph theoretical, probabilistic, and region based methods. Therefore, we can extract structural information in the image to verify object appearance. We also developed novel measures to grade the degree of urbanization in a region. We also introduced a method to detect damaged buildings. In the remainder of the thesis, we investigate each object detection problem (urban region, building, and road segments) in a separate chapter. We summarized previous studies at the beginning of

each chapter. At the end of each chapter, we present and discuss our experimental results.

2. DETECTING AND GRADING THE URBAN REGION

Automatically detecting and monitoring urban regions is an important problem in remote sensing. In this chapter, we propose automated techniques for the first step of urban monitoring; detecting and grading urban regions. First, we use the well-know SIFT (Scale Invariant Feature Transform) features and a graph theoretical approach to detect urban region boundaries. We test our SIFT based urban region detection algorithm on very high resolution panchromatic Ikonos images. Unfortunately, SIFT based algorithm requires high computation time and it can not be used on our aerial images. Therefore, we next propose a Gabor feature based method to detect urban region boundaries in both Ikonos and aerial images.

Land planning organizations, municipalities, disaster relief and environment protection agencies need to keep track of development in a prespecified region in time. Commercially available very high resolution satellite images are suitable for this purpose. These grayscale images can be acquired in relatively short time intervals. They should be inspected periodically to detect urban region changes. Therefore, we also define novel land development measures based on our Gabor features.

In Section 2.5, we test all our methods on a fairly diverse and representative image set formed of panchromatic Ikonos images of different urban sites in Adana, Ankara, and Istanbul cities and grayscale aerial images of Istanbul city of Turkey. These images represent various urban region characteristics. To test our land development measure algorithm, we used sequential images captured from the same region in different times. Test results indicate the potential use of our urban region detection and land development measurement approaches. Before explaining our novel detection methods in detail, we start investigating previous studies on urban region detection in the literature.

2.1. PREVIOUS STUDIES

In the literature, many researchers proposed solutions to the urban region detection problem using automated techniques. Karathanassi *et al.* [3] used building density information to classify residential regions. They benefit from texture information and segmentation to extract residential regions. Unfortunately, they had several parameters to be adjusted manually. Benedik-

tsson *et al.* [4] used mathematical morphological operations to extract structural information to detect the urban area in satellite images. They benefit from neural networks for classification purposes. Therefore, they need training data to detect urban areas. Ünsalan and Boyer [5, 6, 7, 8] used structural features to classify urban regions in panchromatic satellite images. Since they use statistical classifiers, they also need training data to detect the urban region in the image. In a following study, Ünsalan and Boyer [9] associated structural features with graph theoretical measures in order to grade satellite images and extract residential regions from them. Fonte *et al.* [10] considered corner detectors to obtain the type of structure in a satellite image. They concluded that, corner detectors may give distinctive information on the type of structure in an image. Bhagavathy and Manjunath [11] used texture motifs for modeling and detecting regions (such as golf parks, harbors) in satellite images. They focused on repetitive patterns in the image. Bruzzone and Carlin [12] proposed a context-based system to classify very high resolution satellite images. They used support vector machines fed with a novel feature extractor. Fauvel *et al.* [13] fused different classifiers to extract and classify urban regions in panchromatic satellite images. Zhong and Wang [14] extracted urban regions in grayscale satellite images using a multiple classifier approach. These last three studies also need a training data for urban region classification. Zerubia *et al.* [15] used texture information to detect the urban region in both optical and radar images. They extracted texture parameters using chain based Gaussian models. They clustered these with a Markovian segmentation step. Their system is robust to sensor changes, but not very robust to resolution changes in imagery. Doğrusöz and Aksoy [16] detected building clusters using multispectral information. After constructing a Voronoi graph by assuming building centers as vertices of this graph, they classified residential regions as ordered or disordered. Gamba *et al.* [17] used boundary information to extract the map of an urban region. They fed the boundary and non-boundary data to two different classifiers. Then, they combined results of two classifiers. They obtained satisfactory results to detect urban region buildings on VHR imagery. Yang and Newsam [18], compared SIFT and Gabor filtering based features to classifying remotely sensed images. They concluded that Gabor features perform better than SIFT based features for classification purposes. The literature is vast on this topic. Some related papers can be counted as [19, 20, 21, 22, 23, 24].

There are fairly mature change detection algorithms in the literature [25, 26]. Roysam *et al.* [27] offer a detailed literature search on change detection methods. Although most change detection methods are powerful, they have two main deficiencies. First, these methods can not grade the change in time. Second, most of the change detection algorithms require perfect align-

ment between the two images to detect changes. To overcome perfect alignment problem, Li *et al.* [28] used Scale Invariant Feature Transform (SIFT) features to detect changes in urban objects. They first partitioned the image, then analyzed changes at a subgraph level. In a similar manner, Tang *et al.* [29] proposed a framework for structural change detection analysis using remotely sensed image pairs. They analyzed similarities and differences by calculating the distance of local features between the two images. The more advanced setting is not only detecting changes but also measuring them. Ünsalan [30], used graph theory to grade changes in urban regions.

2.2. URBAN REGION DETECTION USING SIFT DESCRIPTORS AND GRAPH THEORY

In this section, we propose a novel method based on SIFT features and graph theoretical tools to detect the urban region boundaries from very high resolution panchromatic Ikonos images. Before extracting SIFT features, we first upsample the image by six to help SIFT feature extraction step. Then, we apply a nonlinear bilateral filtering (BF) operation to smooth out unwanted noise terms in the image. However median filter can also smooth out unwanted noise terms without damaging edges, bilateral filtering operation is stronger to smooth out texture on objects. By smoothing texture with bilateral filtering, we can obtain more generic SIFT features which does not depend on a certain object. Then, we extract local SIFT features from the processed satellite image. Lowe [31] proposed Scale Invariant Feature Transform (SIFT) for object detection based on its template image. It has valuable properties such as invariance to illumination and viewpoint. In the literature, it is extensively used to match objects, represented by template images, in a given image [32, 33, 34, 35]. Mikolajczyk and Schmid [36] compared SIFT descriptors with other invariant feature descriptors. They concluded that SIFT performs best under changes in scale, rotation, and illumination. Interestingly, recent research in neuroscience has shown that object recognition in primates makes use of features of intermediate complexity that are largely invariant to changes in scale, location, and illumination as SIFT features [31].

Unfortunately, the standard SIFT implementation is not sufficient for urban region detection from satellite images. Standard SIFT algorithm is designed for detecting only one specific object in given test image. To do so, each feature in the model image is matched with only one feature in the test image which is most similar. In our problem, we try to detect building clusters in the test image. These buildings have similar properties, and they are located nearby.

Therefore, standard feature matching does not work properly. First, we develop standard SIFT algorithm for multiple matching. Then, we cast the urban region detection problem as one of multiple subgraph matching. In forming the graph, each SIFT feature is taken as a vertex. The neighborhood between different vertices is summarized as edges of the graph. Using multiple subgraph matching, we detect urban region boundaries. Next, we describe the preprocessing step.

2.2.1. Bilateral Filtering for Preprocessing

To detect the urban region, we should first eliminate noise and very small (redundant) details in the satellite image. Unfortunately, objects are so small in these images that, it is almost impossible to discard noise terms without disturbing object boundaries using standard linear filters. Therefore, we propose using bilateral filtering proposed by Tomasi and Manduchi [37]. This filter performs nonlinear smoothing on images by keeping the edge information. In bilateral filtering, nonlinear smoothing is performed by combining the geometric and intensity similarity of pixels. We next explain the bilateral filtering operation using Elad's notation [38].

Let $I_g(x, y)$ be a grayscale image having values in the range $[0, 1]$. Let $I_b(x, y)$ be the bilateral filtered version of $I_g(x, y)$. The filtering operation can be represented as

$$I_b(x,y) = \frac{\sum_{n=-N}^{N} \sum_{m=-N}^{N} W(x,y,n,m) I_g(x-n, y-m)}{\sum_{n=-N}^{N} \sum_{m=-N}^{N} W(x,y,n,m)} \tag{2.1}$$

This equation is simply a normalized weighted average of a neighborhood of $2N+1$ by $2N+1$ pixels around the pixel location (x, y). The weight $W(x, y, n, m)$ is computed by multiplying the following two factors as

$$W(x,y,n,m) = W_s(x,y,n,m) \times W_r(x,y,n,m) \tag{2.2}$$

where $W_s(x, y, n, m)$ stands for the geometric weight factor. It is based on the Euclidean distance between the center pixel (x, y) and the $(x - n, y - m)$ pixel as

$$W_s(x,y,n,m) = \exp\left(-\frac{(x-n)^2 + (y-m)^2}{2\sigma_s^2}\right) \tag{2.3}$$

This way, nearby samples influence the final result more than distant ones.

The second weight, $W_r(x, y, n, m)$ is based on the grayscale intensity distance between the values of the center pixel (x, y) and the $(x - n, y - m)$ pixel. Again, it is based on the Euclidean distance between intensity values as

$$W_r(x, y, n, m) = \exp\left(-\frac{(I_g(x, y) - I_g(x - n, y - m))^2}{2\sigma_r^2}\right) \tag{2.4}$$

Thus, pixels with close grayscale intensity values tend to influence the final result more than those having distant values.

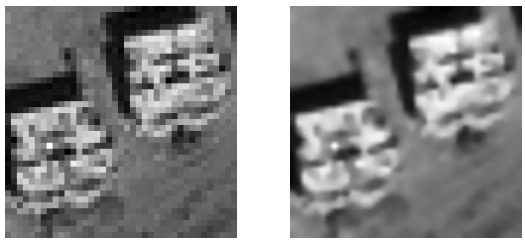
The bilateral filter is controlled by three parameters. N dictates the support of the filter. Larger support gives stronger smoothing. The parameters σ_s and σ_r control the decay of the two weight factors. We pick $N = 5$, $\sigma_s = 3$ and $\sigma_r = 0.1$ for implementation. These values are picked keeping in mind the minimum geometric size and intensity variations of building in the Ikonos image to be detected. For other satellite or aerial image types, these parameters should be adjusted accordingly.

We use the $Adana_8$ image (a typical test image) given in Fig. 2.1 to illustrate our system step by step. In this test image, it is hard to detect the urban region and the buildings due to the low contrast between building rooftops and the background. To provide a detailed explanation

Figure 2.1. The $Adana_8$ test image

of our methods, we focus on a subpart of this image as in Fig. 2.2. Here, we only consider two buildings. We first provide the bilateral filtering results of this subpart image in the same figure. As can be seen, the bilateral filtered image is fairly smooth, with the edge information of

buildings kept fairly well.



(a) Subpart image (b) Bilateral filtering

Figure 2.2. Subpart of the $Adana_8$ test image, and its bilateral filtering result

2.2.2. Scale Invariant Feature Transform (SIFT)

SIFT leads to extracting local features and descriptor vectors, invariant to translation, scaling, and rotation. They are also partially invariant to illumination changes and 3D projection. In this method, first a template image is obtained for the object to be detected in the test image. Its feature locations and descriptor vectors are extracted. Then, feature locations and descriptor vectors for the test image are extracted. A one to one matching between template and test image descriptors lead to detecting the object in the test image.

Buildings can be considered as objects in the satellite image to be detected. However, in satellite images buildings can have different illumination conditions, structural differences and size changes. Most importantly, they can be imaged from different viewpoints. The invariance properties of SIFT can handle most of these variations. Therefore, SIFT is suitable for building detection from satellite images. The following are the major steps to generate SIFT features and their descriptor vectors from an image. To note here, more detail on SIFT operations can be found in [31]. Here, we briefly summarize this method only for our purposes.

2.2.2.1. Scale Space Analysis

The first stage of feature detection is to identify locations and scales that can be repeatedly assigned under differing scales of the same object. Detecting locations that are invariant to scale change of the image can be accomplished by searching for stable features across all possible scales, using a continuous function of scale known as scale space. The scale space of an image is defined as a function $L(x, y, \sigma)$ that is produced from the convolution of a variable scale

Gaussian, $G(x, y, \sigma)$ with the input image $I_b(x, y)$ as

$$L(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) * I_b(x, y) \tag{2.5}$$

where $*$ is the convolution operation in x, y and σ is the Gaussian scale.

2.2.2.2. Feature Localization

To efficiently detect stable feature locations in scale space, Lowe proposed using the scale-space extrema. The extrema is calculated from the difference of Gaussian images computed from the difference of two nearby scales separated by a constant multiplicative factor k as

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{2.6}$$

In order to detect the local maxima and minima of $D(x, y, \sigma)$, each sample point is compared to its eight neighbors in the current image and nine neighbors in the scale above and below. It is selected only if it is larger than all of these neighbors or smaller than all of them. Once a feature candidate has been obtained by comparing a pixel to its neighbors, the next step is to perform a detailed fit to the nearby data for location, scale, and ratio of the principal curvatures. This information allows points to be rejected that have low contrast (and are therefore sensitive to noise) or are poorly localized along the edge. For details of this operation, see [31]. We label the feature by its spatial coordinates as (x_i, y_i).

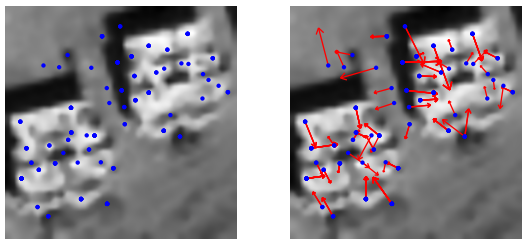
2.2.2.3. Orientation Assignment

One or more orientations are assigned to each feature location based on local image gradient directions. By assigning a consistent orientation to each feature based on local image properties, the feature descriptor can be represented relative to this orientation. Therefore, it has an invariance to image rotation. To obtain orientation information, an orientation histogram is formed from the gradient orientations of sample points within a region around the feature (also known as support region). The orientation histogram has 36 bins covering the 360 degree range of orientations. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian weighted circular window with a σ that is 1.5 times that of the scale of the feature. Peaks in the orientation histogram correspond to the dominant directions of local gradients. The

highest peak in the histogram is detected, and then any other local peak that is within 80% of the highest peak is used to also create a feature with that orientation.

2.2.2.4. Feature Descriptor

The feature descriptor generated by SIFT algorithm is created by sampling the magnitudes and orientations of the image gradient in the patch around the feature, and building smoothed orientation histograms to capture the important aspects of the patch. A 4×4 array of histograms, each with 8 orientation bins, captures the rough spatial structure of the patch. This 128 element vector is then normalized to unit length to have an invariance to illumination. It is thresholded to remove elements with small values. We represent this 128 element feature descriptor as $\mathbf{f_i}$. We extract features and vector representations from the $Adana_8$ subpart image as given in Fig. 2.3. As can be seen, features in this image are located around building corners and intensity changes.



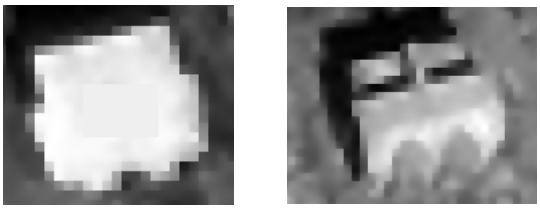
(a) Feature locations (b) Vectors

Figure 2.3. SIFT feature locations and their vector representation on the $Adana_8$ subpart image

2.2.3. Detecting the Urban Region

In the original SIFT proposed by Lowe, the object to be detected in the image is represented by one or several template images. Then, descriptor vectors are obtained for each template. Features for the test image are also obtained. A one to one matching between template features and the test image features is performed using the Euclidean distance between descriptor vectors. SIFT descriptors are highly discriminative. Therefore, each template feature matches with the test image feature if they really resemble. This is the strength of the original SIFT based object detection. However, this property of the SIFT is not suitable for our problem. First, we have many buildings in the satellite image to be detected. Second, it is not feasible to have templates for all types of buildings to be detected. Therefore, we propose a novel graph theoretical method to detect the urban region boundaries.

To detect the urban region, we use two template building images as given in Fig. 2.4. The first template represents a bright building, having a high contrast between background and its rooftop. The second template represents a dark building, having relatively low contrast between the background and its rooftop. These two templates cover a wide range of building characteristics considering our test image dataset. Our template buildings are capable to detect building characteristics in our test images taken from three different cities in Turkey, but they may not perform well in images which contains very different kind of building structures. In this case, template buildings should be chosen manually from the test region. As mentioned before, each



(a) Bright building (b) Dark building

Figure 2.4. Two building templates used in this study

SIFT feature is described by a vector $\mathbf{v}_i = (x_i, y_i, \mathbf{f}_i)$. Here, (x_i, y_i) represent the spatial coordinate of the feature. $\mathbf{f_i}$ is the 128 element feature vector for that feature. We first extract features for the two templates and the test image. Then, we represent them as $\mathbf{v}_i^a$ $i = 1...I$ for the dark building template; $\mathbf{v}_j^r$ $j = 1...J$ for the bright building template, and $\mathbf{v}_m^t$ $m = 1...M$ for the test image respectively.

2.2.3.1. Graph Representation

To detect the urban region and buildings, we cast the problem in terms of graph theory. A graph G is represented as $G = (V, E)$, where V is the vertex set and E is the edge matrix showing the relations between these vertices. For the urban region and building detection, we represent features extracted from the dark building template ($\mathbf{v}^a$), bright building template ($\mathbf{v}^r$), and test image ($\mathbf{v}^t$) in a graph formation as $G^a(V^a, E^a)$, $G^r(V^r, E^r)$, and $G^t(V^t, E^t)$ respectively. Let's consider G^a. $V^a = \{\mathbf{v}_i^a\}$ $i = 1...I$. E^a is an $I \times I$ matrix defined as

$$E^a(i,j) = \begin{cases} d_{ij} & \text{if } d_{ij} < \epsilon_1 \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

where $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$. We take $\epsilon_1 = 30$ due to the size of buildings we are detecting. Also, $E^a(i,j) = 0$ means that there is no edge between vertices $\mathbf{v}_i$ and $\mathbf{v}_j$. We form G^r and G^t in a similar way.

2.2.3.2. Multiple Subgraph Matching to Detect the Urban Region

Buildings in a region indicate the existence of an urban region there. Therefore, in order to detect the urban region, it is sufficient to detect buildings. To detect all buildings in an image (without selecting a special one), we apply multiple subgraph matching between G^a, G^t and G^r, G^t separately. Applying multiple subgraph matching between the template and test images is different from the original SIFT. As mentioned above, we have many buildings in the same region and we want to detect all at once without selecting a specific one. This formalism simplifies our urban region detection problem. The main reason is that, nearby buildings effect each other's detection probability in multiple subgraph matching. Also, using this formalism, we relax the graph matching condition. This is also the case in actual life. A building in a region gives hint about a nearby building, since humans favor cluster of buildings to settle. In other saying, we first detect building clusters. Then we detect each building from this cluster. The important point here is that, detecting one building positively affects detecting nearby buildings.

From now on, we will explain our multiple subgraph matching method on the G^a, G^t pair. The method is the same for the G^r, G^t pair. Our multiple subgraph matching method can be rephrased as a one to many matching between two graph vertices both in unary and binary terms. For the unary matching between G^a and G^t, we define multiple vertex matching between two graphs $G^a = (V^a, E^a)$ and $G^t = (V^t, E^t)$ as

$$\mathbf{M_1}(\mathbf{v}_i^a, \mathbf{v}_j^t) = \begin{cases} 1 & ||\mathbf{f}_i^a - \mathbf{f}_j^t|| < \epsilon_2 \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

$\mathbf{M_1}(\cdot,\cdot)$ will be an $I \times M$ matrix. We check matching $\forall \mathbf{v}_i^a \in V^a$ and $\forall \mathbf{v}_j^t \in V^t$. We limit the multiple matching number to 50, taking into account the average number of buildings in a test image. In Eqn. 2.8, ϵ_2 stands for the tolerance value for unary matching. After extensive tests on our test image database, we set $\epsilon_2 = 4$.

For urban region and building detection, matched features should also keep their structure.

Therefore, we define binary vertex matching between two weighted graphs $G^a = (V^a, E^a)$ and $G^t = (V^t, E^t)$ as

$$\mathbf{M_2}(E^a(i,j), E^t(k,l)) = \begin{cases} 1, & if(\mathbf{M_1}(v_i^a, v_k^t) = 1) \\ & \wedge(\mathbf{M_1}(v_j^a, v_l^t) = 1) \wedge (\gamma < \epsilon_3) \\ 0, & \text{otherwise} \end{cases} \tag{2.9}$$

where $\gamma = |E^a(i,j) - E^t(k,l)|$. Here, ϵ_3 is the tolerance value for binary matching. Based the dimension of buildings in or building template images, we set $\epsilon_3 = 4$.

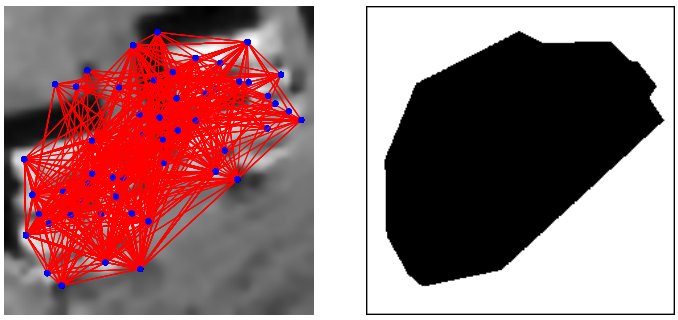
We form a new graph, based on unary and binary matching (matched vertices and edges of G^t). We call it as $G^d = (V^d, E^d)$ to indicate that it contains vertices and edges of the test image graph matched with the dark building template graph. $\mathbf{v}_m^t \in V^d$ iff $\exists \mathbf{v}_i^a \in V^a$ such that $\mathbf{M_1}(\mathbf{v}_i^a, \mathbf{v}_m^t) = 1$. We form the edge matrix E^d as

$$E^d(k,l) = \begin{cases} E^t(k,l) & \text{if } \exists\, i,j \text{ st. } \mathbf{M_2}(E^a(i,j), E^t(k,l)) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

$$E^d(k,l) = \begin{cases} E^t(k,l) & \text{if } \exists\, i,j \text{ st.} \\ & \mathbf{M_2}(E^a(i,j), E^t(k,l)) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.11}$$

We provide the constructed graph G^d on the $Adana_8$ subpart image in Fig. 2.5 (a). As can be seen, all the matched vertices of G^t (now the vertices of G^d) lie on the buildings in the image. Due to multiple subgraph matching, most vertices on different buildings also have edges connecting them. We apply the same procedure to form $G^b = (V^b, E^b)$ in the same way using the G^r, G^t pair. This graph indicates the unary and binary matching between the bright building template and the test image.

To locate urban region containing buildings, we define *region of a graph*. For a graph



(a) Graph, G^d (b) Region, $A(G^d)$

Figure 2.5. The graph and its region obtained from the $Adana_8$ subpart image

$G(V, E)$ with vertices $V = \{v_i\}$ having spatial coordinates $v_i = (x_i, y_i)$, we define its region $A(G)$ as

i- $v_i \in V \Rightarrow (x_i, y_i) \in A(G)$

ii- Let l_{ij} be the line segment joining v_i, v_j where $E(i, j) \neq 0$; $(x_a, y_a) \in l_{ij} \Rightarrow (x_a, y_a) \in A(G)$

iii- Let t_{ijk} be the triangle with corners $v_i, v_j, v_k \in V$ where $E(i, j) \neq 0$, $E(i, k) \neq 0$, $E(j, k) \neq 0$; $(x_a, y_a) \in t_{ijk} \Rightarrow (x_a, y_a) \in A(G)$

This region is as small as possible. It includes all vertices and line segments joining them. To clarify this operation, we formed the region of graph G^d on the $Adana_8$ subpart image. We provide $A(G^d)$ for this image in Fig. 2.5 (b).

There may be both bright and dark buildings in a given test site. Therefore, we detect the final urban region, R using both $A(G^d)$ and $A(G^b)$ as

$$R = A(G^d) \bigcup A(G^b) \tag{2.12}$$

We provide the detected urban region for the $Adana_8$ test image in Figure 2.6. As can be seen, the urban region in this image is correctly detected. We provide more urban region detection examples, as well as qualitative results in Section 2.5. Next, we describe a different approach that we developed to solve the same urban region detection problem.

Figure 2.6. Detected urban region for the $Adana_8$ test image

2.3. URBAN REGION DETECTION USING GABOR FEATURES

In the previous section, we introduced a novel method to detect urban regions in very high resolution panchromatic satellite images. This method performs very well on our diverse image test set. Unfortunately, the method has three main deficiencies. First, the detection performance depends on the chosen building templates. If the region includes unusual structure types, the user should prepare a new building model database. Second, extraction of SIFT features and multiple subgraph matching is computationally costly. Third, SIFT descriptor vectors are generated using small support regions around extracted feature locations. Therefore, it is not perfectly suitable for images in JPEG format (such as our aerial images).

To overcome these shortcomings, we propose a novel method to detect urban regions in this section. Different from our SIFT based approach and most of the studies in the literature, this novel approach does not need any training data for urban region detection. Instead, we developed novel local features based on Gabor filtering and spatial voting. Our method is able to detect urban regions as long as buildings are dense. Besides, we do not have any other constraints. To provide an experimental justification to our method, we tested it on diverse aerial and Ikonos satellite images. We obtained encouraging results.

2.3.1. Local Feature Point Extraction

We benefit from Gabor filters to extract spatial building characteristics (such as edges and corners) in different orientations. Before extracting Gabor features we smooth our test image by median filtering [39]. This step eliminates small noise terms in the image. Then, we apply Gabor filtering in different directions. The maxima in these filter responses lead to local feature points.

Then, we form a voting matrix using them. Finally, we label the urban region in given image by an optimal adaptive decision making approach. Next, we explore these steps in detail.

2.3.1.1. Gabor Filtering

Gabor features are widely used in various important computer vision tasks such as texture segmentation and face recognition [40, 41, 42, 43]. Gabor filter responses can exhibit such desirable characteristics of spatial locality and orientation selectivity since they are localized in the space and frequency domains optimally [44].

Mathematically, the two dimensional Gabor filter can be defined as the product of a Gaussian and a complex exponential function as

$$F_{\varphi}(x,y) = \frac{1}{2\pi\sigma_g^2} \exp\left(-\frac{u^2+v^2}{2\sigma_g^2}\right) \exp\left(j2\pi f u\right) \tag{2.13}$$

where $u = x\cos\varphi + y\sin\varphi$ and $v = -xsin\varphi + y\cos\varphi$. f is the frequency of the complex exponential signal, φ is the direction of the Gabor filter, and σ_g is the scale parameter. These parameters should be adjusted with respect to the image resolution at hand. In this study, we explore the effect of these parameters in Section 2.5.2.1. In Fig. 2.7, we represent real part of Gabor filter response in spatial domain for $\varphi = 0$ filtering direction. We can detect edge-

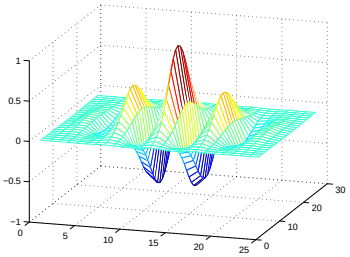


Figure 2.7. Real part of Gabor filter in spatial domain for $\varphi = 0$ filtering direction

oriented urban characteristics in a test image, $I_g(x,y)$ having size $N \times M$, using Gabor filtering. Therefore, we benefit from the real part of the Gabor filter response as

$$G_{\varphi}(x,y) = \mathfrak{Re}\{I_g(x,y) * F_{\varphi}(x,y)\} \tag{2.14}$$

$G_{\varphi}(x,y)$ is maximum for image regions having similar characteristics with the filter. In the next

section, we use this information to extract local feature points.

2.3.1.2. Local Feature Points

To extract local feature points, we first search for the local maxima in $G_\varphi(x, y)$ for $x = 1, ..., N$ and $y = 1, ..., M$. If any pixel (x_o, y_o) in $G_\varphi(x, y)$ has the largest value among its neighbors, $G_\varphi(x_o, y_o) > G_\varphi(x_n, y_n) \; \forall \; (x_n, y_n) \in \{(x_o - 1, y_o - 1), (x_o, y_o - 1), ..., (x_o + 1, y_o + 1)\}$; we call it as a local maximum. It is a candidate for being a local feature point. Next, we check the amplitude of the filter response $G_\varphi(x_o, y_o)$. We call our local maximum (x_o, y_o) as a candidate local feature point if and only if $G_\varphi(x_o, y_o) > \alpha$. To handle different images, we obtain α using Otsu's method on $G_\varphi(x, y)$ in an adaptive manner for each image separately [45]. Therefore, we eliminate weak candidate local feature points in future calculations.

To represent each candidate local feature point further, we assign a weight, w_o, to it as follows. We first threshold $G_\varphi(x, y)$ with α and obtain a binary image $B_\varphi(x, y)$. In this image, pixels having value one correspond to strong responses. We obtain connected pixels to (x_o, y_o) in $B_\varphi(x_o, y_o)$. By definition, two pixels are connected (in a binary image) to each other if there is a path (of pixels with value one) connecting them [39]. As we obtain all the connected pixels to (x_o, y_o), we assign their sum as the weight w_o. Therefore, if a candidate local feature point has more connected pixels, it has more weight. We expect the candidate local feature points to represent urban characteristics such as building clusters. Unfortunately, all candidate local feature points may not represent reliable information on the urban region. Therefore, we discard candidate local feature points having weight, w_o, less than 20 pixels. Finally, we obtain the local feature points for the given φ direction.

We apply this procedure in all φ directions and obtain a total of K local feature points as (x_k, y_k) with their weights w_k for $k = 1, ..., K$. We expect these local feature points to be located on building edges in the image. We provide such an example on the sample $Adana_6$ satellite test image in Fig. 2.8. As can be seen, most local feature points are located on the building edges in this image. In the literature, there are also more complex feature point extraction methods [36]. However, for urban region detection we do not need perfect local point extraction from the image. Since we use their ensemble, missing a few local feature points does not affect the performance of our urban region detection method. We explore urban region detection using Gabor filtering based local feature points next.

Figure 2.8. The $Adana_6$ test image, local feature points extracted by Gabor filtering

2.3.2. Detecting the Urban Region

As we obtain Gabor filtering based local feature points, the next step is detecting urban regions using them. Therefore, we form a voting matrix based on spatial voting first. Then, we apply an optimum decision making method to detect the urban region from the voting matrix. We explain these steps in detail next.

2.3.2.1. Voting Matrix Formation

To detect an urban region, we should have many local feature points in it. These should also be closely located in spatial domain. Therefore, we define a voting matrix based on extracted local feature points as follows. We assume that, around each local feature there is a high possibility of an urban characteristic (such as a building). Therefore, each local feature has the highest vote at its spatial coordinate (x_k, y_k) and its votes decrease with respect to spatial distance. Based on this definition, we form the voting matrix for $x = 1, ..., N$ and $y = 1, ..., M$ as

$$V(x,y) = \sum_{k=1}^{K} \frac{1}{2\pi\sigma_k^2} \exp\left(-\frac{(x-x_k)^2 + (y-y_k)^2}{2\sigma_k^2}\right) \tag{2.15}$$

where σ_k is the parameter for voting proximity for each local feature. For our test images, we pick $\sigma_k = 5 \times w_k$ to add some tolerance for voting. If σ_k has higher values, each local feature will have a wider spatial effect in the voting matrix. Therefore, the false alarms in urban region detection will increase. On the other hand, if σ_k has lower values, each local feature will have a narrower spatial effect. Hence, the correct urban region detection results will decrease.

For the $Adana_6$ test image, we provide the voting matrix in Fig. 2.9. In this figure, vote values are color coded (red corresponds to the highest and blue to the lowest vote value). As can be seen, votes are cumulated around buildings.

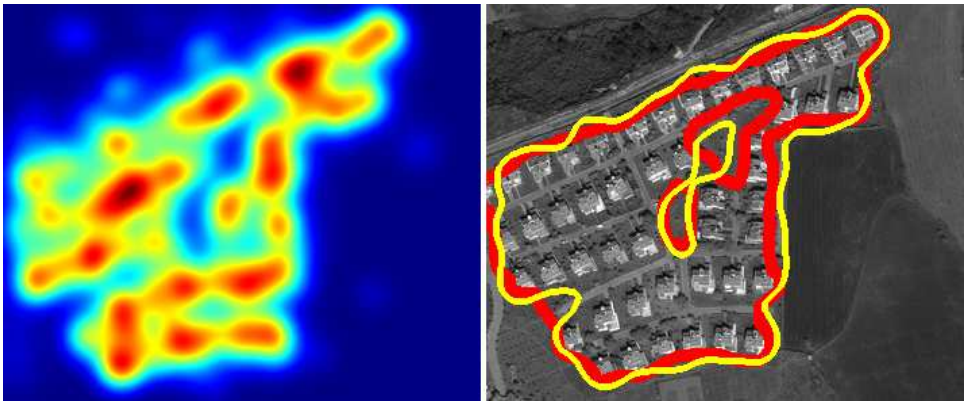


Figure 2.9. The $Adana_6$ test image, the voting matrix obtained and the urban region detected (with ground truth data)

2.3.2.2. Optimum Decision Making

Locations with high votes in $V(x, y)$ are possible urban region pixels and locations with low votes are possible non-urban region pixels in the image. Therefore, we expect to have a bimodal pixel distribution on $V(x, y)$ (one peak corresponds to the urban region and the other to non-urban region pixel votes). We use this information and Otsu's method to detect the urban region [45]. Based on Bayes decision criteria, Otsu's method finds the optimal threshold level (assuming Gaussian probability density functions) between urban and non-urban pixel votes. In this study, the threshold value is adaptively obtained for each test image separately. Since this operation is adaptive, we do not need any manual (or predefined) parameter adjustments.

For the $Adana_6$ satellite test image, we provide the voting matrix pixel distribution in Fig. 2.10. As can be seen in this distribution, there are two peaks (one corresponding to urban and the other to the non-urban region votes). In the same figure, we also provide the optimum decision boundary obtained by Otsu's method by a red dashed vertical line. As can be seen, the decision boundary is correctly located. Based on this threshold value, we provide the detected urban region (as a yellow curve) from the $Adana_6$ image in Fig. 2.9. In the same figure, we also provide our ground truth data for the $Adana_6$ test image as a thick red curve. As can be seen, our detection result closely fits to the ground truth data. Besides thresholding with Otsu's method, we also consider the overall votes in the test image. If their accumulation is not sufficient, we assume

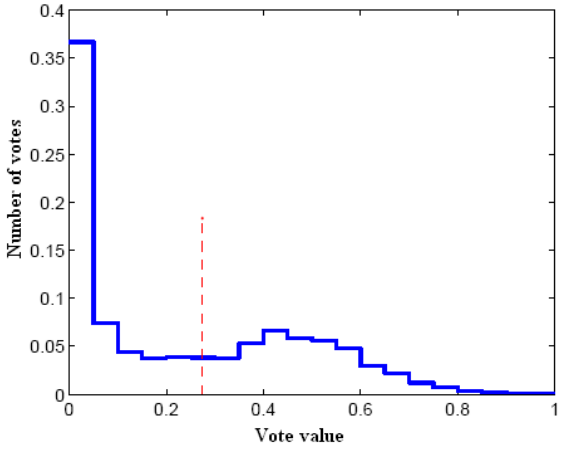


Figure 2.10. The voting matrix pixel distribution and the optimum threshold value for the $Adana_6$ test image

they can not represent an urban region. To perform this test on an image basis, we assume that if the detected urban region size does not exceed 5% of the image size, there is no urban region there. After extensive testing, we obtained this value for our test images. However, it is not strict and can be changed in a relaxed manner.

To summarize, our method depends on local feature point extraction using Gabor filtering. We use these local feature points in forming a spatial voting matrix. Then by using an optimum decision making approach, we are able to detect the urban region in a given aerial or satellite image. After extensive testings, we obtained very encouraging results with our method. We present the performance of our algorithm on test images in Section 2.5.

2.4. GRADING THE URBAN REGION USING LOCAL FEATURES

To grade the degree of development on a given region, we benefit from Gabor filtering based features that we described in Section 2.3. We represent them in a spatial voting matrix. Based on this representation, we define five novel land development measures on a sequential image set acquired from the same location. In all our land development measures, we do not need perfect alignment (or preregistration). We assume that user extracted interested urban regions manually in sequential satellite images without trying to correspond spatial coordinates, and rotations of images. In fact, this is the main strength of our method. Our method also does not depend on the image size in the sequence. Therefore, scales of the images can slightly change due to the satellite's looking angle. We test our method on 19 sets of sequential panchromatic Ikonos

images. Our test results indicate the possible use of our method in measuring land development automatically. We will present these results in Section 2.5.

2.4.1. Local Feature Extraction and Representation

In Section 2.3.1, we explained Gabor feature extraction in detail. Here, we benefit from same local Gabor features to measure land development in sequential satellite images. As in Section 2.3.2.1, we represent these local Gabor features in a voting matrix to be used in measuring land development.

2.4.2. Measuring Land Development

As we obtain the voting matrix (possibly representing the human activity in a given satellite image), we introduce five novel land development measures on it. These measures are designed to summarize the voting matrix in various ways. We measure the development in a given urban region (with its sequential images in time) as follows. We calculate the specified land development measure for each image in the sequence. We expect these measures to indicate development either by increasing or decreasing with respect to time. Next, we introduce each novel land development measure in detail.

2.4.2.1. Number of Voting Local Features

Our first land development measure is based on the total number of voting local features in a given image. As mentioned above, we designed Gabor filters to detect possible man-made structures such as buildings. Local features are based on these filter responses. Therefore, in a given image if the number of man-made structures increase, we expect to have more local features. Hence, we define our first land development measure as the total number of local features as $m_1 = K$.

2.4.2.2. Normalized Sum of Votes

As we mentioned above, the voting matrix is formed by the votes of local features in the given image. For more developed regions, we expect to have more votes in the voting matrix. The reason for this is twofold. First, we will have more local features in more developed regions. Second, we expect each local feature in a developed region to have more votes. Therefore, the

sum of votes in the voting matrix may represent the development. We obtain our second land development measure (m_2) on an $N \times M$ image as

$$m_2 = \frac{1}{NM} \sum_{x=1}^{N} \sum_{y=1}^{M} V(x, y) \tag{2.16}$$

We apply normalization by the image size, since all images in the sequence may not have the same size.

2.4.2.3. Maximum Vote

Related to the normalized sum of votes measure, we can also benefit from the maximum vote from the voting matrix to measure land development. In the normalized sum of votes measure, we take all the votes in the voting matrix into account. However, this may be misleading for some development regions (such as the ones having occluded buildings). Therefore, we define the maximum vote value as our third land development measure (m_3) as

$$m_3 = \max_{(x,y)} V(x, y) \tag{2.17}$$

2.4.2.4. Normalized Urban Area

We also measure the land development by extracting the possible urban region in a given image. Our hypothesis is that, as the development increases in a region, the urban region also grows there. At this point, the voting matrix helps us to detect the possible urban region in a given satellite image. We first explain how to obtain the threshold value to detect the urban region in the sequence. Assume that, we have T sequential images from a region. Therefore, we have T voting matrices (each separately corresponding to an image). As we mentioned in Section 2.4.2.2, more votes possibly correspond to more developed regions. We have this information as our second measure, the normalized sum of votes, m_2. Since we have T images in the sequence, we have m_2^t, for $t = 1, ..., T$. We take the maximum m_2^t as our benchmark and obtain the optimum threshold value for the corresponding voting matrix using Otsu's method [45]. Let's call this threshold value α. For each image (having size $N \times M$) in the sequence, we calculate

our fourth land development measure (m_4) as

$$m_4 = \frac{1}{NM} \sum_{x=1}^{N} \sum_{y=1}^{M} (V(x,y) > \alpha) \tag{2.18}$$

2.4.2.5. Normalized Sum of Votes in the Urban Region

We finally measure the development using the normalized sum of votes in the urban region. In the previous section, we explored how to extract the possible urban region in the given image. We also based our fourth land development measure by the normalized urban area as in Eqn. 2.18. Here, we extend this measure by also adding normalized sum of votes in the extracted urban region. Then, our fifth land development measure (m_5) for an $N \times M$ image becomes

$$m_5 = \frac{1}{NM} \sum_{x=1}^{N} \sum_{y=1}^{M} (V(x,y) > \alpha) \times V(x,y)) \tag{2.19}$$

where the multiplication operation is on a pixel basis in Eqn. 2.19. Using this formulation, we want to benefit from both the area and sum of votes in the urban region.

2.4.2.6. Fusion of Features

In a previous study, Ünsalan [30] presented a new method to fuse extracted measures in order to increase the performance. Besides using our five novel land development measures, we also fuse them using Ünsalan's fusion method to improve the performance. In this way, we obtain a single quantity to measure development. In fusion, we normalize each feature to [0, 1] by a hard limiting function. We also normalize our features in terms of their decrease or increase in time. That is, some of our features may decrease to indicate development, and some others may increase. If we fuse them without without any preprocessing, then they may inhibit each other. To overcome this problem, we multiply the feature by minus one if it is decreasing to indicate development. We fuse our features by taking their mean values. However there are more detailed fusion techniques in literature, simply taking their mean values is adequate in our application.

2.4.3. Sample Results

To explain our land development measures, we pick two representative urban regions ($Adana_1$ and $Adana_4$) as given in Fig. 2.11. The first urban region, $Adana_1$, corresponds to a building construction zone. This region is a bare land in the first image. In the second and third images in the sequence, the buildings become visible. We expect our land development measures to automatically indicate this activity. The second urban region, $Adana_4$, corresponds to a parkland formation. In the first image, this region is also a bare land. In the second and the third images in the sequence, the development is visible. In Fig. 2.11, we also provide the fusion of land development measures for each test image sequence (in the first and second rows) separately. As can be seen in Fig. 2.11, the fusion of land development measures correctly indicates

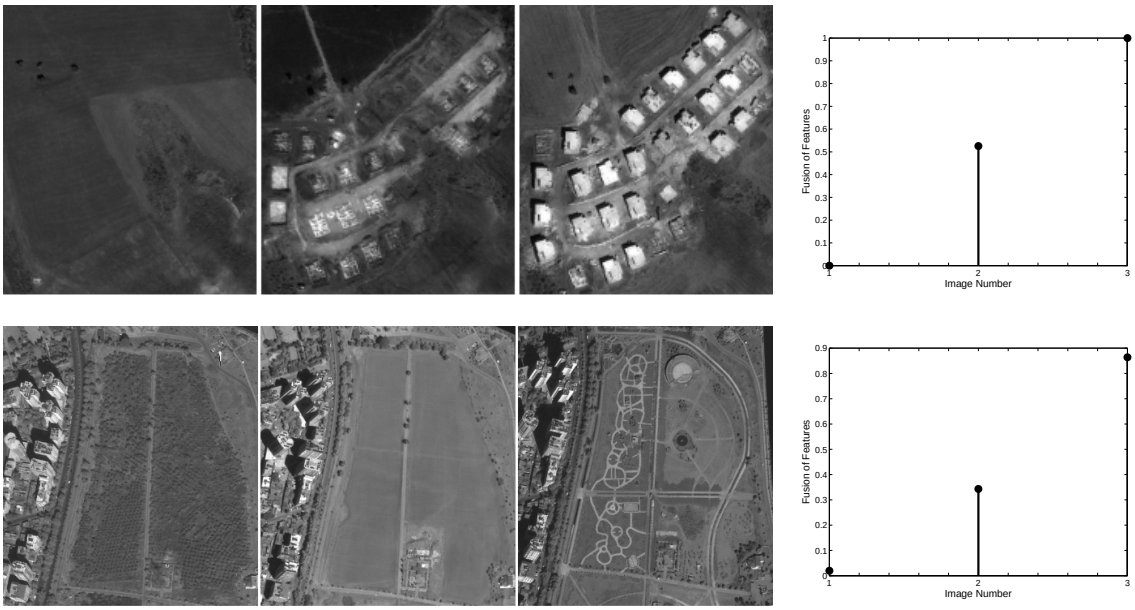


Figure 2.11. Test sequences of $Adana_1$ (first row) and $Adana_4$ (second row) and fusion of land development measures calculated on them

the development in both image sequences by increasing its value. Therefore, we have a perfect performance on these two image sequences. On the $Adana_1$ test image sequence, all other measures also performed perfect. However, for the $Adana_4$ image sequence, measures m_1 and m_3 could not perform perfect. Both measures missed one development step and assigned it as less developed compared to the previous image in the sequence. The reason for this shortcoming is possibly the type of development activity there.

2.5. EXPERIMENTAL RESULTS

In this section, we present experimental results of our urban region detection systems. After representing results on real test images and tabulating their performances with tables, we compare important features and deficiencies of the proposed systems. Next, we start with presenting experimental results of SIFT and graph theory based urban region detection system.

2.5.1. Urban Region Detection Using SIFT Descriptors and Graph Theory

We test our urban region detection method on 30 very high resolution panchromatic Ikonos images. These images are acquired from different sites in Adana, Ankara, and Istanbul cities in Turkey. In these test images, the size and shape of buildings in the region, their proximity, environment, and contrast with respect to background all differ. These test images are specifically selected to represent a wide and diverse urban region characteristics. We provide our test images in Figs. 2.12, 2.13, 2.14, and 2.15 in the first columns. We provide the detected urban region for each test image in the second columns. Next, we analyze these detection results quantitatively.

Our method detects the urban region for each test image fairly well. To quantify these detection results, we formed the ground truth for each test image to the best of our knowledge. In forming the ground truth, we label a region as urban, if it contains building clusters. We provide the urban region detection performances (in percentages) for all test images in Table 2.1. In this table, TP stands for correct urban region detection ratio) FA stands for false urban region detection ratio. The size of each image and the number of urban region pixels (in the ground truth image), labeled as 'UA' is also given in this table.

On a total of 30 test images (with 775714 pixels of total urban region), we obtain a $\mathbf{89.52}\%$ correct urban region detection and a $\mathbf{8.02}\%$ false alarm rate. For our diverse test set, this result is very promising. Table 2.1 can give us further information on our urban region detection method. The lowest urban region detection rate is obtained on the $Adana_{12}$ image. Here, buildings are sparse. Therefore, the region does not show strong urban characteristics. In other saying, there are many non-urban pixels surrounding building clusters. Since our urban region detection method depends on building cluster detection, this poor performance is reasonable. However, the false alarm rate for this image is fairly low. The highest false alarm rate is obtained on the $Adana_2$ image. The reason for this high false alarm rate is that some nearby regions are also

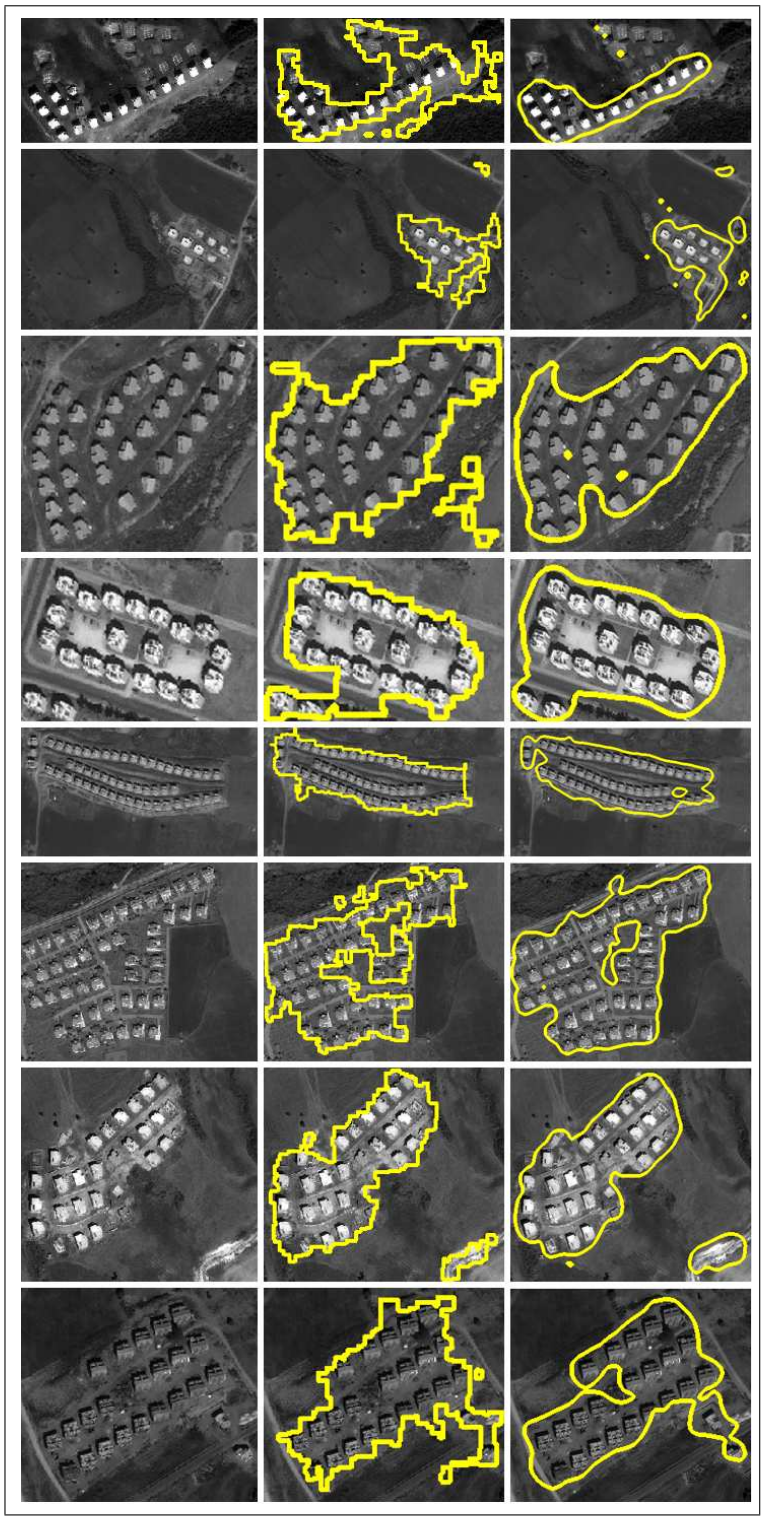


Figure 2.12. Urban region detection results for *Adana* images (1 to 8) for each row separately. First column: test images; second column: detected urban regions with SIFT based algorithm; third column: detected urban regions with Gabor based algorithm

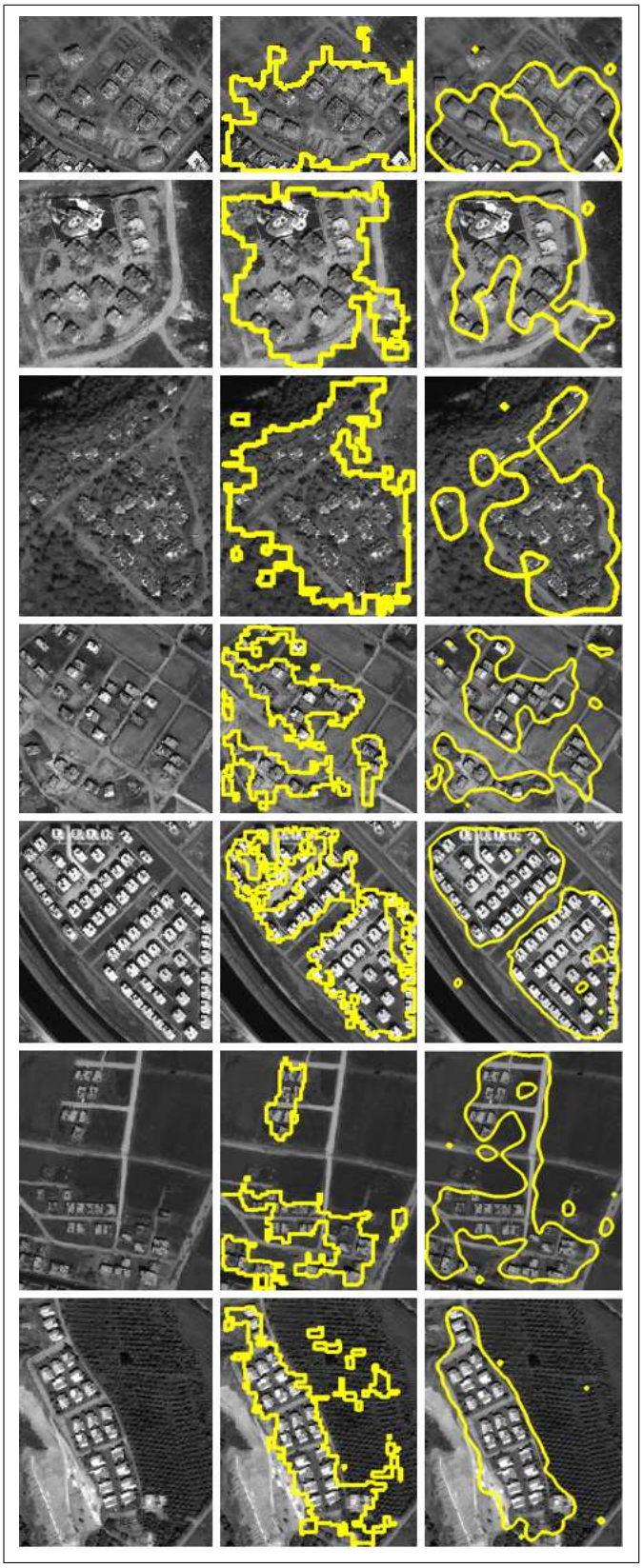


Figure 2.13. Urban region detection results for *Adana* images (9 to 15) for each row separately. First column: test images; second column: detected urban regions with SIFT based algorithm; third column: detected urban regions with Gabor based algorithm

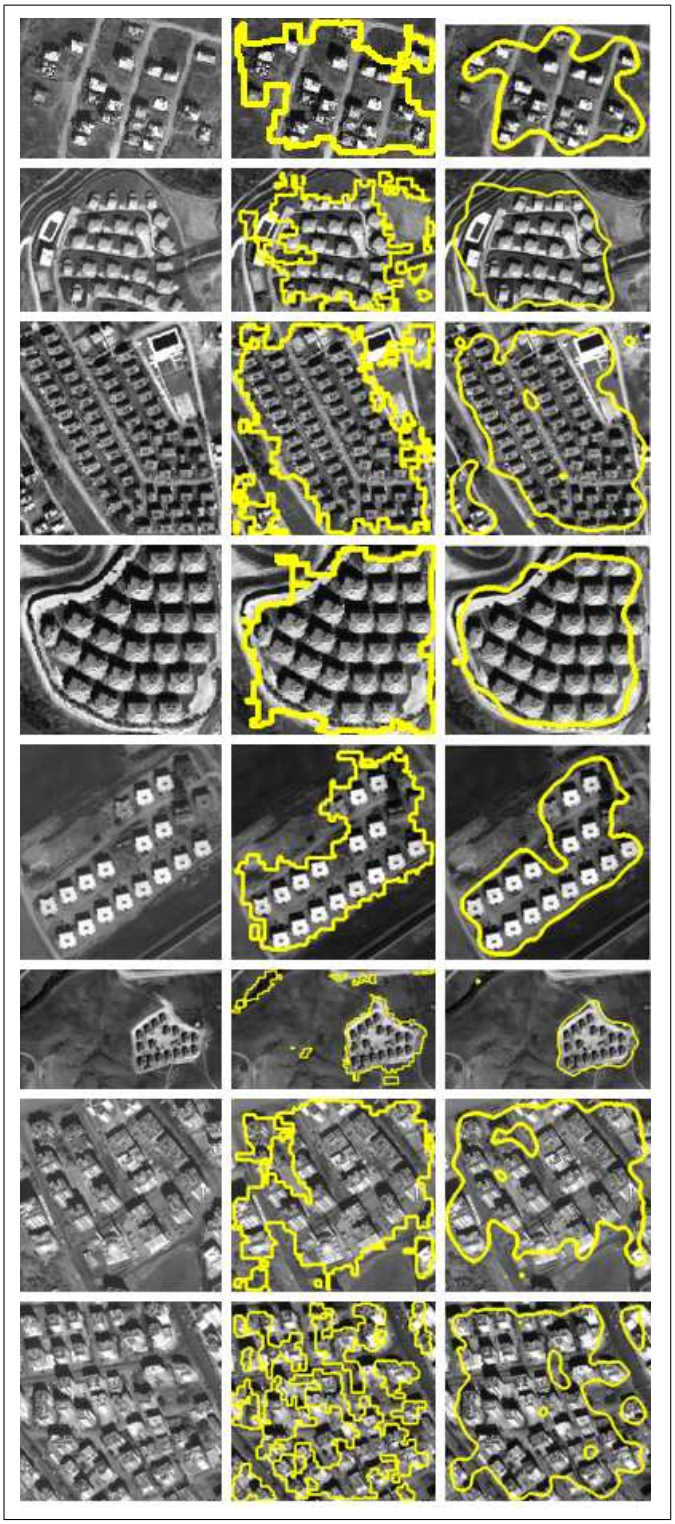


Figure 2.14. Urban region detection results for *Adana* images (16 to 23) for each row separately. First column: test images; second column: detected urban regions with SIFT based algorithm; third column: detected urban regions with Gabor based algorithm

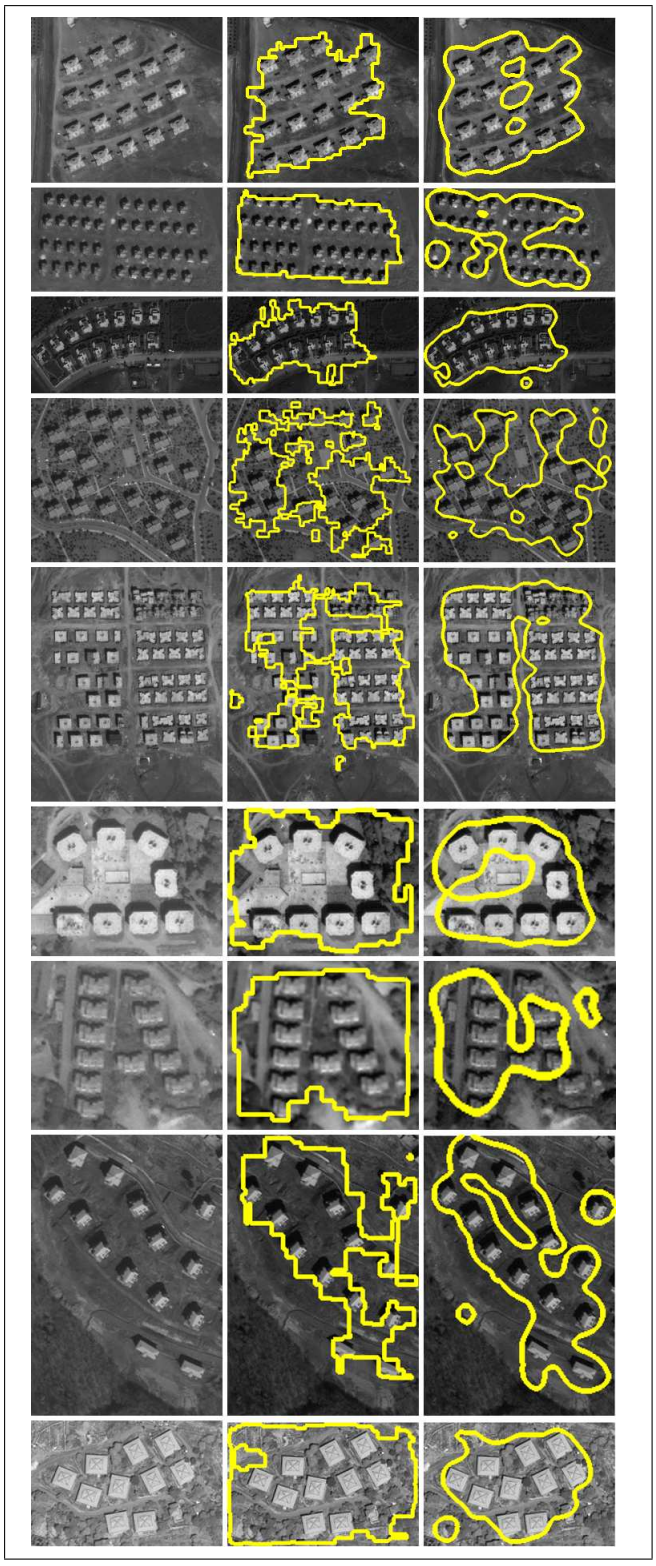


Figure 2.15. Urban region detection results for $Ankara$ images (1,5) and $Istanbul$ images (1 to 4) for each row separately. First column: test images; second column: detected urban regions with SIFT based algorithm; third column: detected urban regions with Gabor based algorithm

Table 2.1. Urban region detection performances (in percentages) for test images with SIFT based approach.

Image Name	**Size (pixels)**	**UA (pixels)**	**TP**(%)	**FA**(%)
$Adana_1$	166×318	17848	98.06	22.42
$Adana_2$	302×401	9735	90.96	39.48
$Adana_3$	192×213	23283	94.87	5.62
$Adana_4$	143×210	15035	99.26	9.86
$Adana_5$	242×450	36058	91.84	1.55
$Adana_6$	293×354	35805	94.83	8.77
$Adana_7$	289×324	31276	95.40	7.64
$Adana_8$	235×265	28747	84.26	1.62
$Adana_9$	166×208	25094	89.40	1.05
$Adana_{10}$	192×200	22016	94.78	13.77
$Adana_{11}$	228×186	25332	95.39	6.06
$Adana_{12}$	257×267	37810	76.88	1.53
$Adana_{13}$	325×289	53482	85.30	0.54
$Adana_{14}$	336×275	26033	84.37	3.30
$Adana_{15}$	334×264	26288	86.81	15.30
$Adana_{16}$	119×173	14517	97.04	2.41
$Adana_{17}$	216×306	29242	87.62	12.69
$Adana_{18}$	258×247	47031	92.42	5.22
$Adana_{19}$	171×185	22382	98.05	8.13
$Adana_{20}$	222×210	19108	87.88	12.90
$Adana_{21}$	265×447	17343	87.34	2.43
$Ankara_1$	208×240	25226	90.36	3.92
$Ankara_2$	130×239	20037	94.08	1.07
$Ankara_3$	145×285	16986	95.13	15.61
$Ankara_4$	271×315	40513	84.60	13.16
$Ankara_5$	340×278	48769	81.69	1.36
$Istanbul_1$	128×162	13266	97.49	17.84
$Istanbul_2$	98×111	6141	99.69	31.95
$Istanbul_3$	248×169	21241	73.70	4.42
$Istanbul_4$	127×193	12725	99.49	57.82
Total		**775714**	**89.52**	**8.02**

taken as urban. This is not totally wrong, since they also show some degree of urban characteristics. The best urban region detection rate is obtained on the $Adana_4$ image. In this test site, buildings are regularly spaced and highly distinctive. Therefore, the urban region in this image is detected with a very high TP.

2.5.1.1. Tests on Different Modules

Our urban area and building detection methods are composed of many submodules (such as bilateral filtering, SIFT keypoint extraction, and building template matching). Therefore, in this section we test the effect of these different submodules on the final detection results. We pick the $Adana_8$ test image for this purpose. First, we test the effect of the bilateral filter. There is also a faster version of bilateral filtering proposed by Paris and Durand [46]. We call this filtering as 'Fast BF'. Instead of the normal bilateral filter (we call it as 'Normal BF'), we use this fast implementation. Second, we test the effect of the building templates for detection. Instead of using two building templates (as dark and bright), we test using each template alone. Based on these variations, we provide the urban area detection results in Table 2.2.

Table 2.2. Urban area detection results on the $Adana_8$ image; SIFT based approach, parameter variations.

	Normal BF		**Fast BF**	
Template	$TP(\%)$	$FA(\%)$	$TP(\%)$	$FA(\%)$
Bright	77.82	1.63	74.05	0.61
Dark	70.25	0.01	64.93	0.34
Both	84.26	1.62	80.85	0.82

As can be seen in Table 2.2, in detecting the urban area, the dark building template has the lowest performance. Using both building templates drastically improves the performance. Using normal or fast bilateral filtering has a significant effect in urban area detection performance.

2.5.1.2. Computation Times

In this part, we tabulate the time needed to detect urban region boundaries. To note here, timing directly depends on the test image. As the number of buildings in a test image increases, the number of local features will also increase. Therefore, the descriptor vector and multiple subgraph matching algorithms will need more computation times. To give an idea, here we

consider the $Adana_8$ test image as a benchmark. We tabulate all CPU timings for each module in Table 2.3. In this table, we also provide timings for both normal and fast bilateral filtering operations. In reporting these results, we used a PC with an Intel Core2Due processor with a 2.13 GHz clock speed and having 4 GB of RAM. We used Matlab as our coding platform except the SIFT implementation. The SIFT package is taken from Lowe's web site, and it is written in C language. If C platform is used to code all modules, a much faster system can be obtained.

Table 2.3. CPU Times (in seconds) for urban region detection operation on the $Adana_8$ test image with SIFT based approach.

Module Template	**Dark**	**Both**
Upsampling	0.19	0.19
Normal BF	62.67	62.67
Fast BF	6.51	6.51
SIFT keypoints	0.19	0.38
Graph matching	12.42	18.73

In Table 2.3, we provide both normal and fast BF implementations. Similarly, we provide the computation times for using only the dark building template and both templates. We can summarize the different scenarios as follows. Using normal BF and both templates, urban region detection operation requires 81.97sec. This scenario is for obtaining the best performance for urban region detection. If we can tolerate slightly lower detection performances, then we can use fast BF and only the dark template. In this scenario, urban region detection requires only 19.31 sec. The user should select the suitable scenario (both in terms of detection performance and CPU time needed) for his needs.

2.5.2. Urban Region Detection Using Gabor Features

Here, we present experimental results of our Gabor feature based urban region detection method. To test the performance of this method, we use the same panchromatic satellite image data set that we have used to test SIFT and graph theory based algorithm. As a result, we use 30 panchromatic Ikonos satellite images which are specifically selected to represent wide and diverse urban region characteristics. In this part, we also test our urban region detection system on a real aerial image database with 21 aerial test images which have 0.3 m spatial resolution.

On our Ikonos test images, we first provide the experimental justification for selected parameter values. In the following two sections, we provide the overall urban region detection performance of our method on satellite and aerial images separately. We quantify these tests by reporting TP and FA. We also provide the computation times for each operation in our urban region detection method on a sample test image.

2.5.2.1. Tests on Parameter Values

Although we do not have many parameters in our urban region detection method, we provide their effect on the final detection results for satellite images in this section. We first comment on Gabor filter parameters. We observed that, in panchromatic Ikonos images building edges can be represented as a ramp edge with three to four pixels width. Therefore, in this study we picked $\sigma_g = 1.5$, and $f = 0.65$ after extensive testing. To cover differently oriented building edges, we tested different φ values. We provide different settings in Table. 2.4. In these tests, we use 30 Ikonos images to eliminate image dependency.

Table 2.4. The effect of Gabor filtering directions on urban region detection.

Number of φ	**TP**(%)	**FA**(%)
4	88.39	11.55
6	88.63	8.21
8	88.47	6.30
10	**89.33**	**5.91**
12	88.79	5.91
14	90.79	5.99

As can be seen in Table. 2.4, the number of Gabor filtering directions have effect on decreasing FA. Based on this test, we can conclude that choosing ten different directions for Gabor filtering ($\varphi = \{0, \pi/10, 2\pi/10, ..., 9\pi/10\}$ radians) is suitable for urban region detection. To note here, choosing six, eight or twelve directions also provide similar TP and FA values. Therefore, we can conclude that our method does not specifically depend on the total Gabor filtering direction.

We then test the effect of median filtering on the final detection result. We provide tests with different median filter sizes in Table. 2.5. Again, in these tests we use 30 Ikonos images to

eliminate image dependency.

Table 2.5. The effect of median filtering on urban region detection.

Median filter size	**TP**(%)	**FA**(%)
No filtering	95.04	31.76
3×3	94.13	16.75
$\mathbf{5 \times 5}$	**89.33**	**5.91**
7×7	84.29	5.15

As can be seen in Table. 2.5, without median filtering FA is fairly high. Therefore, we need a median filtering operation. As can be seen, a 5×5 median filtering gives reasonable TP and FA values.

For aerial images, we also use the same settings. However, to make the resolution of the satellite and aerial images similar, we downsampled aerial images by 0.5. Next, we discuss the overall urban region detection performance of our method on satellite images.

2.5.2.2. The Overall Performance on Satellite Images

The overall performance of our method on 30 satellite images having 775714 urban region pixels is $TP = 89.33\%$ and $FA = 5.91\%$. Therefore, our method was able to detect 692926 urban region pixels correctly with 45854 false alarm pixels. This is a fairly good urban region detection result on such a diverse satellite image set. We provide performance for each satellite test image in Table 2.6 and detection results in the last columns of Figs. 2.12, 2.13, 2.14, and 2.15. We also provide a sample test image for urban region detection results for satellite images in Fig. 2.16. As can be seen in this figure, all urban regions with different building characteristics are correctly detected.

2.5.2.3. The Overall Performance on Aerial Images

Next, we discuss the overall performance of our method on 21 aerial images. In this test, we have 1688936 urban region pixels. We obtained $TP = 85.93\%$ with a $FA = 10.94\%$ for urban region detection. Therefore, 1451225 urban region pixels are correctly detected, with a 184851 pixels of false alarm. Similar to satellite images test, this result on such a data set is fairly good. We provide performance for each satellite test image in Table 2.7 and detection

Table 2.6. Urban region detection performances on satellite images (in percentages) with the Gabor feature based algorithm.

Image Name	**Size (pixels)**	**UA (pixels)**	**TP**(%)	**FA**(%)
$Adana_1$	166×318	17848	65.77	6.84
$Adana_2$	302×401	9735	94.22	27.53
$Adana_3$	192×213	24541	89.44	0.29
$Adana_4$	143×210	15035	98.78	8.41
$Adana_5$	242×450	36058	94.03	3.96
$Adana_6$	293×354	37350	98.92	17.36
$Adana_7$	289×324	31276	95.71	9.47
$Adana_8$	235×265	28747	86.56	0.70
$Adana_9$	166×208	25094	77.86	0.00
$Adana_{10}$	192×200	22016	82.40	10.02
$Adana_{11}$	228×186	25332	83.63	2.65
$Adana_{12}$	257×267	37810	76.36	2.68
$Adana_{13}$	325×289	53482	95.71	1.33
$Adana_{14}$	336×275	26033	85.43	8.85
$Adana_{15}$	334×264	26288	91.71	6.79
$Adana_{16}$	119×173	14517	81.99	1.25
$Adana_{17}$	216×306	33375	94.33	6.85
$Adana_{18}$	258×247	47031	88.10	4.88
$Adana_{19}$	171×185	22382	92.82	2.85
$Adana_{20}$	222×210	19108	89.49	4.81
$Adana_{21}$	265×447	17343	99.12	8.50
$Ankara_1$	208×240	25226	94.81	4.17
$Ankara_2$	130×239	20037	91.19	4.38
$Ankara_3$	145×285	16986	90.34	6.98
$Ankara_4$	271×315	40513	86.20	11.55
$Ankara_5$	340×278	48769	93.81	5.54
$Istanbul_1$	128×162	13266	79.45	1.27
$Istanbul_2$	98×111	6141	86.63	2.13
$Istanbul_3$	248×169	21241	82.96	4.54
$Istanbul_4$	127×193	12725	96.59	9.71
Total		**775714**	**89.33**	**5.91**

Figure 2.16. Urban region detection result on a sample satellite image

results in Figs. 2.17, 2.18, and 2.19. We also provide a sample test image for urban region detection results for aerial images in Fig. 2.20. As can be seen in this figure, all urban regions with different building characteristics are correctly detected.

2.5.2.4. Computation Time

To provide an idea about the computation time needed for our Gabor feature based method, we next consider the CPU timings for each step on the $Adana_8$ satellite test image. We picked this image as a benchmark in our SIFT and graph theory based algorithm in Section 2.2. Therefore, we pick the same image here to compare both methods. In reporting computation times, we used the same PC with Intel Core2Due processor with 2.13 GHz. clock speed and has 4 GB of RAM. We used Matlab as our coding platform. After testing, we obtained that for the $Adana_8$ image Gabor filtering needs 0.61 sec.; local feature point extraction needs 0.94 sec. and spatial voting needs 2.30 sec. The total time needed for urban region detection is 3.85 sec. This is a fairly good timing to obtain the urban region.

2.5.3. Comparison of the Proposed Urban Region Detection Systems

Finally, we compare our urban region detection methods we presented in Sections 2.2, 2.3. In Section 2.2, we introduced a SIFT and graph theory based system to detect urban regions in very high resolution panchromatic satellite images. In 30 satellite images, we obtained an

Figure 2.17. Test results with Gabor feature based algorithm for $Aerial$ images (1 to 7) for each row separately. First column: original test images; second column: detected urban regions

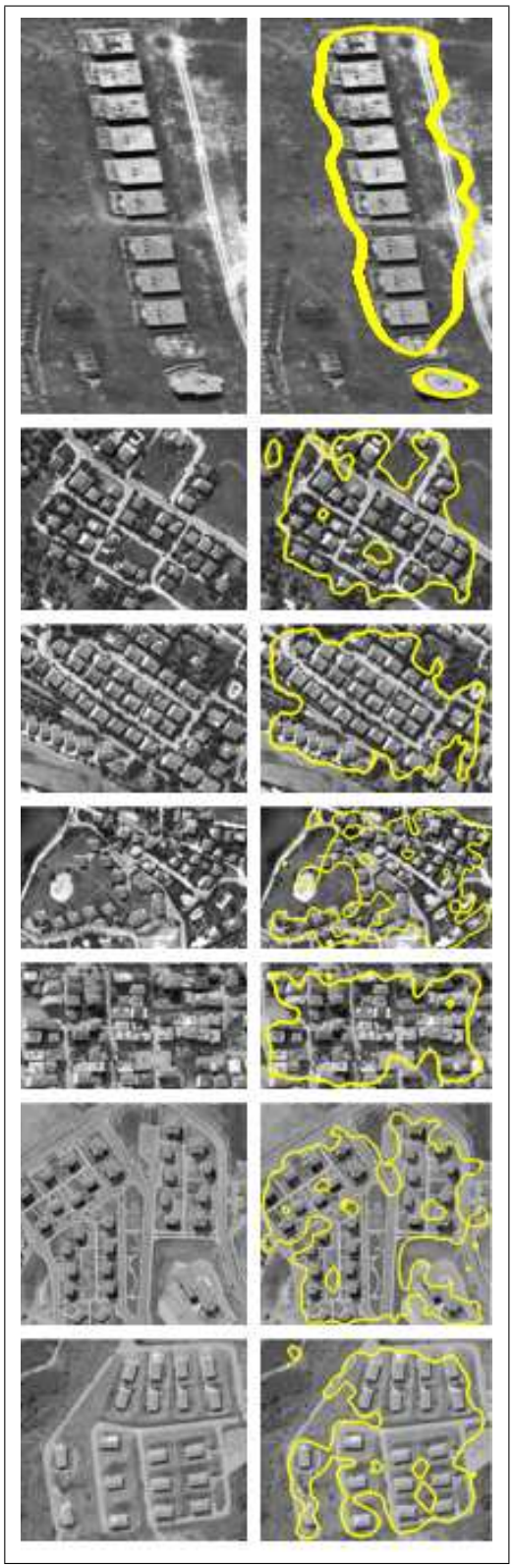


Figure 2.18. Test results with Gabor feature based algorithm for $Aerial$ images (8 to 14) for each row separately. First column: original test images; second column: detected urban regions

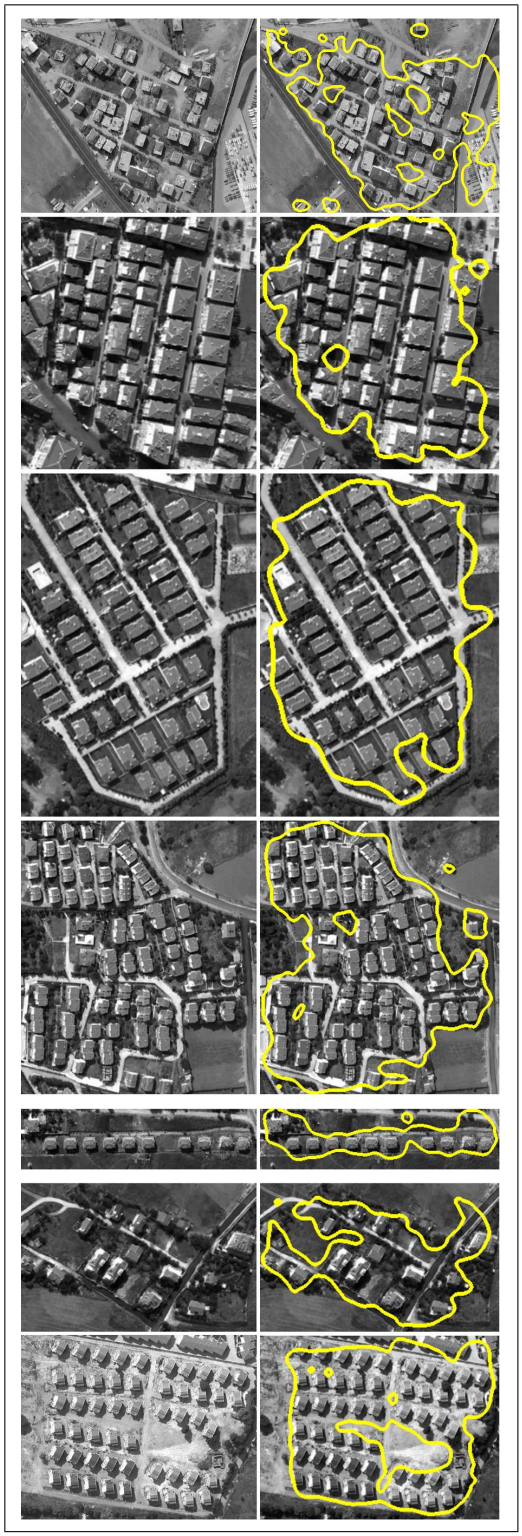


Figure 2.19. Test results with Gabor feature based algorithm for *Aerial* images (15 to 21) for each row separately. First column: original test images; second column: detected urban regions

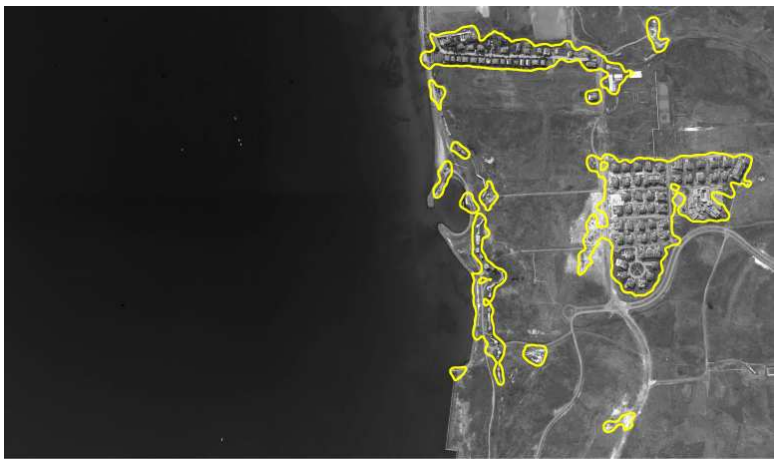


Figure 2.20. Test result with Gabor feature based algorithm for a large aerial image. First row: original test image; second row: detected urban areas

Table 2.7. Urban region detection performance on aerial images (in percentages) with Gabor feature based algorithm.

Image Name	**Size (pixels)**	**UA (pixels)**	**TP**	**FA**	**TP**(%)	**FA**(%)
$Aerial_1$	562×553	43610	41084	22330	94.21	51.20
$Aerial_2$	287×199	11605	11271	3410	97.12	29.38
$Aerial_3$	254×283	17975	14912	4496	82.96	25.01
$Aerial_4$	139×274	17622	9459	2605	53.68	14.78
$Aerial_5$	317×229	32309	29233	7	90.48	0.02
$Aerial_6$	167×219	13745	12175	1629	88.58	11.85
$Aerial_7$	397×117	20282	16423	2331	80.97	11.46
$Aerial_8$	287×357	52831	43526	5284	82.39	10.00
$Aerial_9$	309×497	101772	78852	3693	77.48	3.63
$Aerial_{10}$	373×422	69746	58461	1607	83.82	2.30
$Aerial_{11}$	401×491	71704	63759	17960	88.92	25.05
$Aerial_{12}$	389×271	69719	65407	4080	93.82	5.85
$Aerial_{13}$	434×378	111635	98825	1813	88.53	1.62
$Aerial_{14}$	104×413	18711	16622	2432	88.84	13.00
$Aerial_{15}$	210×336	36022	30214	928	83.88	2.58
$Aerial_{16}$	248×240	28113	25325	4473	90.08	15.91
$Aerial_{17}$	321×301	57248	50785	450	88.71	0.79
$Aerial_{18}$	618×651	200195	175628	998	87.73	0.50
$Aerial_{19}$	1816×3090	355790	315510	53717	88.68	15.10
$Aerial_{20}$	1371×1141	271684	209795	45219	77.22	16.64
$Aerial_{21}$	671×658	86618	83959	5389	96.93	6.22
Total		**1688936**	**1451225**	**184851**	**85.93**	**10.94**

89.52% correct urban region detection performance with an 8.02% false alarm rate. False alarms occurred because of terrain formations resembling buildings (such as sand hillocks) in satellite images. This performance on such a diverse test was noteworthy. Besides this high performance, proposed system had some deficiencies. First, the proposed system needs template buildings which contain specific characteristics of buildings in study region. Second, extracting SIFT features in large satellite images and multiple sub-graph matching operations need very high computation time. In addition to that, computation time increases with number of buildings in given region. Third, SIFT based urban region detection system can not be used on aerial images because of the characteristics of SIFT descriptor vectors. Since our aerial images are saved in lossy Jpeg standard, small details are distorted in these images. This distortion effects support region of SIFT feature descriptors. As a result, descriptor vectors are not generated correctly and detection application could not be performed on our aerial image data set.

In Section 2.3, we introduced the Gabor feature based system. In this system, we used the same data set as in SIFT based system. In 30 satellite images, we obtained $TP = 89.33\%$ with $FA = 5.91$. Comparing the performances with SIFT and graph theory based system, the detection and false alarm rates are almost the same. In Gabor feature based algorithm, we can further apply probabilistic relaxation to improve obtained results. However, we will need extra computations for performing it. Unfortunately, we could not run our SIFT and graph theory based algorithm on aerial image data set. Therefore, here we can not compare performances on aerial image data set.

To compare both methods in terms of CPU timings, we pick the $Adana_8$ satellite test image as a benchmark. The time needed for the SIFT based method was 81.97 sec. With Gabor feature based method, we only need 3.85 sec. Therefore, our Gabor feature based method is fairly fast and almost has the same performance with our SIFT based urban region detection method. Besides, we do not need any template building images (as in the SIFT based method) to detect urban region in Gabor feature based method. Therefore, user can directly run the algorithm on given test image.

2.5.4. Grading the Urban Region Using Local Features

Ünsalan [30] used a data set composed of 19 different urban regions having diverse characteristics for grading land development. Each urban region is imaged either three or four different

times. Therefore, there is a total of 69 images. Here, we also benefited from the same data set to test our new land development measures. We expect our land development measures to indicate the degree of development automatically on these image sets.

2.5.4.1. Overall Performance

We obtain the overall performance of our measures in accordance with the evaluation method given in [30]. There, to assess the performance of a feature, we define an ordinal position for each image in the sequence (from one to three or four) based on the advancement in construction. Next, we sort the image sequence with respect to feature values. We define the error for the image sequence to be $1/2$ of the absolute difference (to avoid double counts) in the ordinal position as assigned by the feature versus that of the ordinal position. The sum of the per-image deviations over the sequence provides the error score for the feature for that image sequence. Finally, we formalize the error as

$$e = \frac{1}{2} min(\sum_{i=1}^{B} |s_o(i) - s_f(i)|, \sum_{i=1}^{B} |s_o(i) - s_f(B-i)|) \tag{2.20}$$

where s_o is the ordinal (desired) sort, and s_f is the sort by the feature. B is the total number of satellite images in given sequence (either three or four in our application). We perform the same calculations for the 19 construction zones separately and obtain the overall error. We obtain the total correct sort value by subtracting the overall error from the total number of images (69 here). Finally, we obtain the percentage of the total correct sort to the total number of images. We indicate it as the average performance of the feature.

We obtain the following overall performance values for m_1 to m_5 as 75.4%, 76.8%, 76.8%, 78.3%, 78.3% respectively. Although the performance values are close to each other, the normalized urban region measure, m_4, and the normalized sum of votes in the urban region measure, m_5, have slightly better performance values. As we fuse all the measures, we obtain an 82.3% performance. Therefore, we gain a 4% performance improvement after fusion.

We provide some sample image sequences and obtained land development measure fusions in Figs. 2.21, 2.22.

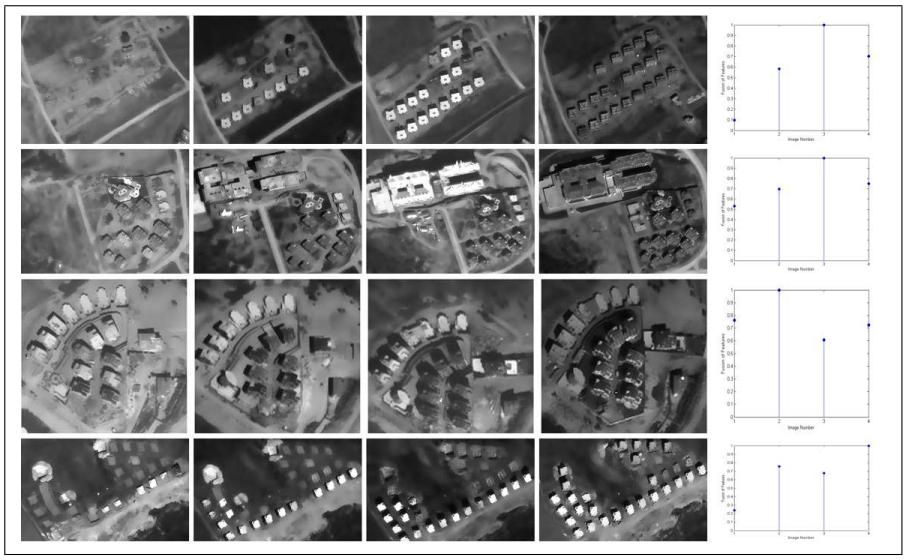


Figure 2.21. Land development test results with Gabor feature based algorithm for $Test$ image sequences (1 to 4) for each row separately. First four column: test images in sequence; fifth column: fusion of land development measures for each image in the sequence

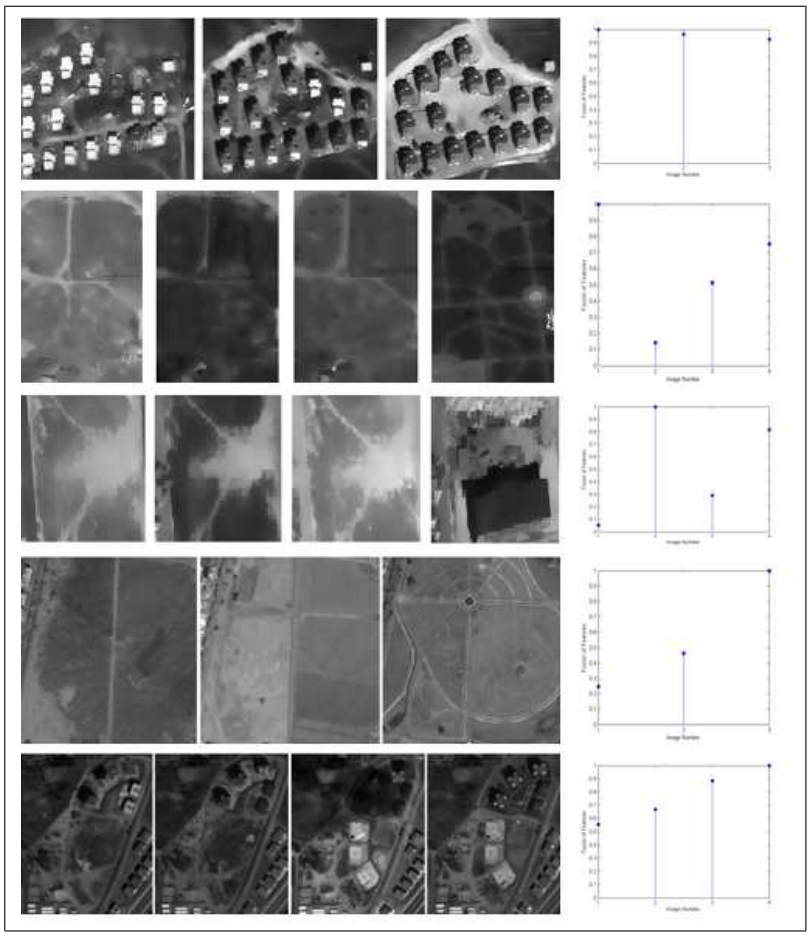


Figure 2.22. Land development test results with Gabor feature based algorithm for $Test$ image sequences (7, 8, 10, 13, 16) for each row separately. First four column: test images in sequence; fifth column: fusion of land development measures for each image in the sequence

2.5.5. Comparison of the Proposed Land Development Measures

In Section 2.4, we introduced a novel approach to detect land development in sequential satellite images based on Gabor features. In Ünsalan's [30] graph theoretical land development measures, the best performance as 72.5% using one measure alone. Our Gabor feature based land development measures introduced in this study have at least 2.9% to 5.8% better performances compared to the graph theoretical ones.

In terms of the computational cost (in terms of CPU times), graph theoretical land development measures need 59.51 sec. on average (over 69 images) with a PC with Intel Core2Due Processor with 2.2 GHz. clock speed. Our Gabor feature based land development measures need 2.17 sec. on average with the same settings. This test indicates that, the computational cost of our new land development measures are also remarkably low compared to the previous method.

2.6. SUMMARY OF THE CHAPTER

In this chapter, we introduced two novel systems to detect urban region boundaries on gray scale very high resolution satellite and aerial images. We also proposed a novel system to grade land development in an urban region using image sequences captured in different times.

First, we introduced novel SIFT feature based method (in Section 2.2). SIFT approach leads to extracting local features and descriptor vectors, invariant to translation, scaling, and rotation. Buildings can be considered as interested objects in the satellite images. Since buildings can have different illumination conditions and their appearance (such as viewing angle) is uncontrolled, SIFT features are suitable for object detection in satellite images. There, we picked two template building images, one representing dark buildings and the other representing bright buildings. We obtain their SIFT features. We also obtain the SIFT keypoints for the test image. Then, by applying multiple subgraph matching between template and test images SIFT features, we detect the urban region in the test image. We test our urban region detection method on a diverse and representative image set. We obtain an 89.52% correct urban region detection performance on such a diverse test set is noteworthy. False alarms in urban region detection mostly occur because of terrain formations resembling buildings (such as sand hillocks) in satellite images. One important feature of the system is that, one can generalize the system to detect any kind of objects in satellite images. However our SIFT based algorithm gives high performance

using only grayscale information, it has some deficiencies. In order to use proposed algorithm, a user should prepare a template building data set which includes buildings showing general building characteristics of the study region. However our algorithm detects urban region boundaries in a few minutes in our test images, calculation time increases exponentially when complexity and size of image increases. As another deficiency we observed that, SIFT based algorithm does not perform well on lossy compressed images (like Jpeg images).

To overcome drawbacks of the SIFT based algorithm, we developed another urban region detection algorithm based on Gabor features. We extracted Gabor features in a given test image using Gabor filters. We use these local features in forming a spatial voting matrix. Then by using an optimum decision making approach, we were able to detect the urban region in given remotely sensed image. After extensive testings, we obtain very encouraging results with our method. Comparing with SIFT based algorithm, we can also conclude that our new urban region detection method is fairly fast and reliable.

Computation time of Gabor feature based algorithm was also very impressive. Also, we do not benefit from multispectral information as in previous studies. We can further apply probabilistic relaxation to improve our results. However, we will need extra computations for performing it. As a further step, we also developed Gabor feature based algorithm to detect urban development. In order to measure development, we used very high resolution panchromatic satellite image sequences of the region. Proposed method depends on the same Gabor features. We used these local features in forming a spatial voting matrix. Then, we defined five different land development measures on it. After extensive testings, we observed that our novel land development measures have similar or better performances compared to previous land development measures in the literature. Besides, the computational cost of our novel land development measures is fairly low compared to previous methods. Therefore, land development measures based on local features (introduced in this study) have their strengths compared to previous ones.

Novel studies presented in this chapter can be taken as the first step in monitoring urbanization. The next chapters will be on detecting separate objects such as buildings and road segments in very high resolution satellite and aerial images.

3. BUILDING DETECTION

Buildings are important objects in remotely sensed images. Therefore, robust detection of buildings is an important part of the automated satellite and aerial image interpretation problem. Automatic detection of buildings enables creation of maps, detecting changes, and monitoring urbanization. However, detecting buildings in satellite and aerial images is a difficult task for several reasons. Building may be imaged from different viewpoints. The illumination and contrast in the image may not be sufficient for detecting buildings. There may be several other structures, such as nearby trees and street segments making the building detection problem harder. In addition to these difficulties, buildings do not have standard size and shape. All these issues make building detection a hard problem.

In this chapter, we propose novel approaches to detect separate buildings in very high resolution satellite and aerial images using local invariant features. To this end, we expand our urban region detection methods that we presented in Chapter 2. After presenting building detection approaches, we propose a novel method to detect damaged buildings and to extract approximate building shapes automatically. At the end of the chapter, we compare performances, beneficial features, and deficiencies of the proposed building detection methods. Next, we present previous work on building detection.

3.1. PREVIOUS STUDIES ON BUILDING DETECTION

In the literature various algorithms are proposed for building detection. Mayer [47], provides an excellent overview of building detection studies developed between 1984 and 1998. These studies generally use gray-scale aerial images and detect the buildings by thresholding and edge detection techniques. Ünsalan and Boyer [5], also provide an extended overview as well as a novel method to detect houses and street segments in residential regions. These overviews may be used to see the overall picture in building detection studies.

Levitt and Aghdasi [48] used the region growing algorithm on grayscale images and segmented most of the buildings in their test images. They verified the existence of buildings using the edge map. Kim and Muller [49], used graph theory to detect buildings in aerial images. They extracted linear features in a given satellite image. They used these features as vertices of a

graph. Then, they extracted buildings by applying subgraph matching with their model building graph. Finally, they used intensity and shadow information to verify the building appearance. Segl and Kaufmann [50] combined supervised shape classification with unsupervised image segmentation in an iterative way. Their method allows searching objects (like buildings) in high resolution satellite images. Mayunga *et al.* [51] used polygons (formed by edge information) to detect buildings. They proposed a novel snake algorithm starting from an approximate polygon center. The snake grows radially until it fits a closed polygon shape. Then, they used linear features to verify the building appearance. The main drawback of this algorithm is its dependence on manually labeled starting points. Peng *et al.* [52] also proposed an improved snake model to detect buildings in color aerial images. They report good detection results with their system.

Huang *et al.* [53] considered fusion of multispectral Ikonos imagery to classify objects (including buildings) in urban regions. Molinier *et al.* [54] considered detecting man-made structures in satellite images using PicSOM. Wei and Xin [55] introduced a method based on level-sets to segment out man-made objects in aerial images. Katartzis and Sahli [56] used a hierarchical stochastic model based on perceptual organization to detect building rooftops in color satellite images. Karantzalos and Paragios [57] developed a novel framework for building detection from grayscale satellite and aerial images. For this purpose, they used level-set segmentation method with multiple shape priors to extract building shapes and poses. Akçay and Aksoy [58] proposed a novel method for unsupervised segmentation and object detection in high-resolution satellite images. They used morphological operators on each spectral band to extract candidate segments. They proposed an object detection algorithm formulating the detection process as an unsupervised grouping problem for the automatic selection of coherent sets of segments corresponding to meaningful structures among a set of candidate segments from multiple hierarchical segmentations obtained from individual spectral bands. Idrissia *et al.* [59] extracted edges of man-made structures (buildings and roads) using Gabor filters together with the NDVI (Normalized Difference Vegetation Index) in SPOT5 images. Comparing their edges of two image sequences taken from same region, they detected changes.

Some researchers used the shadow information to detect buildings. McKeown *et al.* [60] detected shadows in aerial images by thresholding. They showed that building shadows and their boundaries contain important information about building heights and roof shapes. Zimmerman [61] integrated basic models and multiple cues for building detection. He used color, texture, edge information, shadow and elevation data to detect buildings. This algorithm extracts

buildings using blob detection. Tsai [62] compared several invariant color spaces including HSI, HSV, HCV, YIQ, and YC_bC_r models for shadow detection and compensation in aerial images. Vu *et al.* [63] also used shadows to model buildings. They showed that shadow information can be used to estimate damages and changes in buildings. Chen and Hutchinson [64] developed a system to detect damages using bi-temporal grayscale satellite images. First, they compared two images to detect pixel based changes. Then, they extracted object based changes by a probabilistic approach.

Wei and Prinet [65], used a probabilistic framework to detect buildings in high resolution panchromatic satellite images. First, they segmented homogeneous regions using a band pass filter. They assumed these regions as possible building locations. They developed novel structural and textural features and used them to calculate probability of building appearance in this region. Jaynes *et al.* [66] proposed a framework to detect buildings in aerial images. For this purpose, they extracted primitive structures (straight edges) using Boldt algorithm. Using perceptual grouping laws, they detected polygon shapes considering corners and neighbor lines. Unfortunately, this approach is not robust to occlusion and it does not have a step to verify if the polygon shape belongs to a building or not. Mordohai *et al.* [67] used two images of the same region taken with small view angle differences to detect buildings in aerial images. They slid one of these images on another to evaluate the disparity map. Using this disparity map and tensor voting approach, they detected buildings.

In order to generate land maps and discriminate particular buildings, it is also important to detect exact shapes. Shadows in image may help to detect building shapes and they can be used to verify building appearance. Huertas and Nevatia [68] used the relationship between buildings and shadows. They first extracted corners from the image. They labeled these corners as either bright or shadow. They used the bright corners to form rectangles. Shadow corners confirm building hypothesis for these rectangles. McKeown *et al.* [60] showed that building shadows and their boundaries contain important information about building heights and roof shapes. Katartzis and Sahli [56] used shadow information in order to verify wall appearance and to model 3D rooftop models. Krishnamachari and Chellappa [69] introduced a Markov Random Field (MRF) based building detection method in aerial images. They benefit from straight line segments in the image and form their MRF based detection method on their interactions. Wei and Prinet [29] recently introduced a probabilistic method to detect building shapes. They proposed using region information and a probabilistic measure on these to detect man-made structures in the image.

Neuenschwander *et al.* [70] and Rüther *et al.* [71] used snake models to determine exact building shapes from the edge information of remotely sensed images. Although active contour and level set methods provide good detection performances (with building shapes as extra information), as the number of buildings in the image increases their detection time also increases. Besides, occlusions may cause problems for these methods.

3.2. BUILDING DETECTION USING SIFT DESCRIPTORS AND GRAPH THEORY

In Section 2.2, we proposed an urban region detection approach based on SIFT and graph theory. There, we matched SIFT descriptor vectors of the template buildings to SIFT descriptor vectors of test building by our multiple graph matching approach. Then, by applying multiple subgraph matching between template and test image SIFT features we detected a subgraph for test image. We assumed the boundaries of obtained subgraph as urban region boundaries. In this section, we propose a novel graph cut method to detect separate buildings using previously detected urban region subgraph.

3.2.1. Graph Cut Method to Detect Separate Buildings

In Section 2.2, we benefit from the close proximity of buildings in detecting the urban region. In our test image, we obtained two subgraphs G^d and G^b by applying subgraph matching using dark building template and bright building template respectively. Here, our purpose is expanding this approach to detect separate buildings. For building detection, we need extra information. Therefore, we cut some edges of G^d and G^b based on the intensity criteria. We hypothesize that vertices (feature locations) on the same building have similar intensity values due to the building color. Therefore, to locate separate buildings, we cut edges between vertices having different intensity values. We expect the cut edges to be the ones between vertices on different buildings. We form two new graphs as $G^p = (V^p, E^p)$ and $G^q = (V^q, E^q)$. Here, $V^p = V^d$ and $V^q = V^b$. Let's consider G^p, we assign weights to its edges as

$$E^p(k,l) = \begin{cases} 1 & E^d(k,l) \neq 0 \wedge w_{kl} < \epsilon_4 \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

where $w_{kl} = |I_b(x_k, y_k) - I_b(x_l, y_l)|$. (x_k, y_k) and (x_l, y_l) stand for the spatial coordinates of $\mathbf{v}_k, \mathbf{v}_l \in V^p$. $I_b(x, y)$ is our bilateral filtered image. $\epsilon_4 = 0.1$ is the tolerance value (remember $I_b(x, y) \in [0, 1]$). We apply the same procedure to obtain E^q as

$$E^q(k,l) = \begin{cases} 1 & E^b(k,l) \neq 0 \wedge w_{kl} < \epsilon_4 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

The weight assignment step may not work properly if features are located outside the building. To overcome this problem, descriptor vectors may be used to shift each feature location by an offset. For bright buildings (the bright building template is matched with), feature locations are directed outside the building center. For dark buildings (the dark building template is matched with), feature locations are directed towards the building center (as can be seen in Fig. 2.3). This information may be used to shift each feature location inside the building.

Eqns. 3.1 and 3.2 lead to disconnected subgraphs. Each disjoint and connected subgraph possibly represents a building. Therefore, we detect disjoint and connected subgraphs from G^p and G^q. A graph $G(V, E)$ can be decomposed into disjoint and connected subgraphs $G_l(V_l, E_l)$ $l = 1, ..., L$. Each subgraph satisfies the following conditions:

i- $V_l \subseteq V$

ii- $\bigcup_{l=1}^{L} V_l = V$

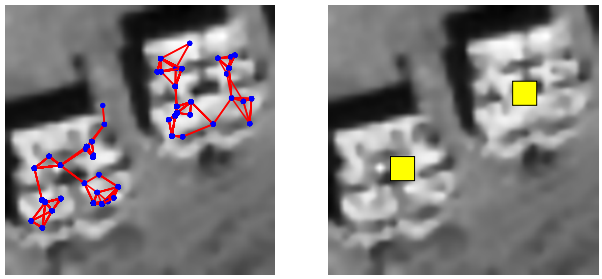
iii- $\forall v_i, v_j \in V_l$ $\exists$ a path between v_i and v_j. A *path* in G is a finite alternating sequence of vertices and edges.

iv-

$$E_l(i,j) = \begin{cases} 1 & \text{if } E(i,j) = 1 \wedge v_i, v_j \in V_l \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

We obtain disjoint and connected subgraphs for G^p using the above definition as G_i^p $i = 1, ..., I$. However, there may be many noise terms. To discard them, we select subgraphs having at least

two vertices. Similarly, we obtain disjoint and connected subgraphs for G^q as G^q_j $j = 1, ..., J$. For the $Adana_8$ subpart image, we provide the obtained disjoint and connected subgraphs G^p_i $i = 1, 2$ in Fig. 3.1. As can be seen, each disjoint and connected subgraph lies on a different building in this test image. As we hypothesized, each subgraph G^p_i $i = 1, ..., I$ and G^q_j $j = 1, ..., J$ represents



(a) Graph cut (b) Buildings

Figure 3.1. Graph cut on the $Adana_8$ subpart image

a candidate building. To locate buildings, we obtain the region of each subgraph as $A(G^p_i)$ and $A(G^q_j)$ separately. Some subgraphs in G^p_i and G^q_j may represent the same building. In other saying, the building may be detected by both dark and bright building templates. To eliminate double counts, we obtain the union of regions as

$$F = A(G^p_i) \bigcup A(G^q_j) \; \forall i, j \tag{3.4}$$

We apply binary labeling on F and obtain its connected components [39]. Each connected component represents a building. If the size of a connected component is less than 1000 pixels we take it to be noise and discard it. This corresponds to approximately a building of size 6×5 pixels in the original image (remember we upsample the test image by six in each coordinate at the beginning). We obtain the center of mass of each connected component and take it as the location of the building it represents.

We provide the building detection result on the $Adana_8$ subpart image in Fig. 3.1. As can be seen, both buildings are correctly detected in this image. We also provide the building detection results on the $Adana_8$ test image in Fig. 3.2. Again, most of the buildings are correctly detected in this test image. Since the contrast between the background and the building rooftops are very low for this image, it is really hard to detect these buildings even for a human observer. We quantify our building detection results on our diverse Ikonos satellite image dataset in Section 3.7. In the next section, we present another novel approach to detect buildings using Gabor

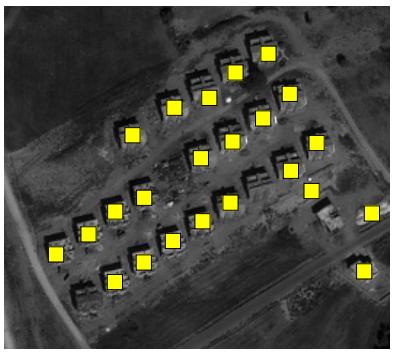


Figure 3.2. Detected buildings in the $Adana_8$ test image

features.

3.3. BUILDING DETECTION USING DIFFERENT LOCAL FEATURES

To overcome difficulties in building detection and to increase accuracy, in this section we propose another novel building detection method using several local feature vectors and a probabilistic framework. We first introduce three different local feature vector extraction methods. Extracted local feature vectors serve as observations of the probability density function to be estimated. Using a nonparametric variable kernel density estimation method, we estimate the corresponding probability density function. In other saying, we represent building locations (to be detected) in the image as joint random variables and estimate their pdf. Using modes of the estimated density, as well as other probabilistic properties, we detect building locations in the image. We also introduce data and decision fusion methods based on our probabilistic framework to detect buildings. We pick our very high resolution panchromatic aerial and Ikonos satellite images to test our method. Extensive tests indicate that our method can be used to automatically detect buildings in a robust and fast manner in satellite and aerial images. We represent detection results in Section 3.7.

3.3.1. Local Feature Vector Extraction

Our probabilistic building detection method depends on local feature vectors extracted from the test image. These are taken as observations of the random building location coordinates (to be estimated). Therefore, we introduce three different local feature vector extraction methods in this section. The first method is based on Harris corner detection. The second method is based on gradient magnitude based support region (GMSR) extraction [72]. Finally, the third method

is based on Gabor filtering in different directions. Next, we explore each method in detail.

3.3.1.1. Harris Corner based Local Feature Vectors

Fonte *et al.* [73] considered (improved) Harris and Susan corner detectors to obtain the type of structure in a satellite image. They concluded that, corner detectors are not sufficient alone to give distinctive information on the type of structure in an image. Schmid *et al.* [74] on the other hand evaluated and compared different corner detectors for general image processing applications. They concluded that the best results are provided by the Harris corner detector [75]. Therefore, we first pick it to extract local feature vectors.

Harris and Stephens define their corner detector (generally known as the Harris corner detector) in three steps: gradient calculation, matrix formation, and eigenvalue calculation. Therefore, first we should calculate smoothed (using a Gaussian function) gradients in x and y directions to detect corners in a given grayscale image $I(x, y)$. We define smoothed gradient filters for x and y directions as

$$g_x(x, y) = \frac{-x}{2\pi\tau_g^4} \exp(-\frac{x^2 + y^2}{2\tau_g^2}) \tag{3.5}$$

$$g_y(x, y) = \frac{-y}{2\pi\tau_g^4} \exp(-\frac{x^2 + y^2}{2\tau_g^2}) \tag{3.6}$$

where τ_g is the smoothing parameter. We take it as unity in this study due to the scale of Ikonos satellite and aerial images at hand. Although our method is fairly robust to this parameter, it should be adjusted by the resolution of the image to be analyzed in future studies.

We calculate the smoothed gradients for the image $I(x, y)$ as

$$I_x = \frac{\partial I(x, y)}{\partial x} = g_x(x, y) * I(x, y) \tag{3.7}$$

$$I_y = \frac{\partial I(x, y)}{\partial y} = g_y(x, y) * I(x, y) \tag{3.8}$$

Harris corner detector depends on calculating a matrix (related to autocorrelation function) as

$$A(x,y) = \begin{pmatrix} a_{xx} & a_{xy} \\ a_{xy} & a_{yy} \end{pmatrix} \tag{3.9}$$

where

$$a_{xx} = \sum_{x_i \in W} \sum_{y_i \in W} I_x^2(x_i, y_i) \tag{3.10}$$

$$a_{xy} = \sum_{x_i \in W} \sum_{y_i \in W} I_x(x_i, y_i) I_y(x_i, y_i) \tag{3.11}$$

$$a_{yy} = \sum_{x_i \in W} \sum_{y_i \in W} I_y^2(x_i, y_i) \tag{3.12}$$

As can be seen, a_{xx}, a_{xy}, a_{yy} are gradient magnitudes averaged over a window W. We pick this averaging window width as seven pixels (due to the resolution of our images) in this study. For further details on this averaging operation, please see [75]. The eigenvalues of matrix A provide information about the edge in a given location. If both eigenvalues of the matrix at a given location is large, then there is a corner there. Harris and Stephens suggested that exact eigenvalue computation can be avoided by calculating the response function

$$R(A) = |A| - \kappa \, trace^2(A) \tag{3.13}$$

where κ is a tunable parameter with values from 0.04 to 0.15 were reported as appropriate in the literature. Therefore, we picked $\kappa = 0.06$ in this study. Harris and Stephens extract their corner points by checking the local maxima of $R(A)$. For a detailed explanation, please see their paper [75].

As we obtain corner points with their spatial coordinates, we define local feature vectors

using them. Besides spatial coordinates, we also add direction and weight information as follows. First, we calculate the gradient orientation, $O(x, y)$, and magnitude, $M(x, y)$, for each image coordinate as

$$O(x, y) = \arctan\left(\frac{I_y(x, y)}{I_x(x, y)}\right) \tag{3.14}$$

$$M(x, y) = \sqrt{I_x^2(x, y) + I_y^2(x, y)} \tag{3.15}$$

For the corner point at coordinate (x_j, y_j), the corresponding orientation is $\theta_j = O(x_j, y_j)$. To assign a weight for the local feature vector, we threshold $M(x, y)$ using Otsu's method for each image separately in an adaptive manner. As a result, we obtain $B_m(x, y)$ as a binary image. In this image, pixels having value one correspond to strong responses. We obtain connected pixels to (x_j, y_j) in $B_m(x_j, y_j)$. By definition, two pixels are connected (in a binary image) to each other if there is a path (of pixels with value one) connecting them. As we obtain all the connected pixels to (x_j, y_j), we assign their sum as the weight w_j. Therefore, if a candidate local feature vector has more connected pixels, it has more weight. Finally, we have Harris corner based local feature vectors as $\vec{k}_h(j) = (x_j, y_j, \theta_j, w_j)$ for $j = 1, ..., K_h$.

3.3.1.2. GMSR based Local Feature Vectors

We next pick a previous study on support region extraction, gradient magnitude based support regions (GMSR) to extract local feature vectors [72]. The method benefits from smoothed gradients to form support regions. Then, these features are used to extract structural and conditional statistical features to classify land use. In this chapter, we extract support regions using smoothed gradient values, I_x and I_y, given in Eqns. 3.7 and 3.8. To extract support regions, we threshold $M(x, y)$ by the $10\ \%$ of the maximum gradient magnitude in the test image. The rationale here is as follows. We take the maximum gradient magnitude as a benchmark. After experiments, we observed that even $10\ \%$ of this value still gives information about the structure in the image. Therefore, we have an adaptive threshold value. Similar to Harris corner detection method, we obtain $B_m(x, y)$ as a binary image after thresholding. In this image, pixels having value one correspond to support regions. For more details, please see [72].

We define local feature vectors based on the extracted support regions. Therefore, we pick each support region pixel as a local feature vector coordinate. Assume that, we have a local feature vector (x_j, y_j). By definition, $B_m(x_j, y_j) = 1$. We define the orientation and magnitude of the local feature vector having spatial coordinate (x_j, y_j) with the same method as we used in the previous section. As a result, we obtain local feature vectors as $\vec{k}_g(j) = (x_j, y_j, \theta_j, w_j)$ for $j = 1, ..., K_g$ from the GMSR.

3.3.1.3. Gabor Filtering based Local Feature Vectors

Finally, we introduce Gabor filtering based local feature vector extraction in this section. In this method, the first step is smoothing the image by median filtering to eliminate small noise terms. Then, we apply Gabor filtering in different directions. Based on these responses, we obtain our local feature vectors.

Mathematically, the two dimensional Gabor filter can be defined as in Eqn. 2.13. As we explained in detail in Section 2.3, f is the frequency of the complex exponential signal, φ is the direction of the Gabor filter, and σ_g is the scale parameter. We observed that, for our test images $\sigma_g = 1.5$, and $f = 0.65$ as suitable values after extensive testing. To cover differently oriented building edges, we tested different φ values. We conclude that choosing ten different directions for Gabor filtering ($\varphi = \{0, \pi/10, 2\pi/10, ..., 9\pi/10\}$ radians) is suitable for building detection.

We can detect building edges and corners in a test image using Gabor filtering. Therefore, for a test image $I(x, y)$ (with size $N \times M$), we benefit from the real part of the Gabor filter response as in Eqn. 2.14. Here, $G_\varphi(x, y)$ is maximum for image regions having similar characteristics with the filter. To extract local feature vector spatial coordinates, we first search for the local maxima in $G_\varphi(x, y)$ for $x = 1, ..., N$ and $y = 1, ..., M$. If any pixel (x_j, y_j) in $G_\varphi(x, y)$ has the largest value among its eight neighbors, $G_\varphi(x_j, y_j) > G_\varphi(x_n, y_n)$ $\forall\ (x_n, y_n) \in \{(x_j - 1, y_j - 1), (x_j, y_j - 1), ..., (x_j + 1, y_j + 1)\}$; we call it as a local maximum. It is a candidate for being a local feature vector coordinate. Next, we check the amplitude of the filter response $G_\varphi(x_j, y_j)$. We call our local maximum (x_j, y_j) as a candidate local feature vector coordinate if and only if $G_\varphi(x_j, y_j) > \alpha$. To handle different images, we obtain α using Otsu's method on $G_\varphi(x, y)$ in an adaptive manner for each image separately. Therefore, we eliminate weak candidate local feature vectors in future calculations.

As we obtain the spatial coordinates of local feature vectors in one Gabor filter direction,

we assign an orientation and weight to them. However, we assign the orientation different than the two previous methods as follows. We check for the orientations in the eight-neighborhood of (x_j, y_j) and pick the orientation, θ_j, as the one having highest magnitude. We applied this procedure to obtain a robust orientation information. We obtain the weight for each local feature vector similar to the methods in previous sections. However, we obtain our binary image $B_m(x, y)$ by thresholding $G_\varphi(x, y)$ with α. We assign a weight to each local feature vector similar to previous sections. The only difference is that we discard local feature vectors having weight larger than 60 pixels. We observed that, these features correspond to non-building segments in the image. We apply this procedure in all φ directions and obtain Gabor filtering based local feature vectors as $\vec{k}_f(j) = (x_j, y_j, \theta_j, w_j)$ for $j = 1, ..., K_f$.

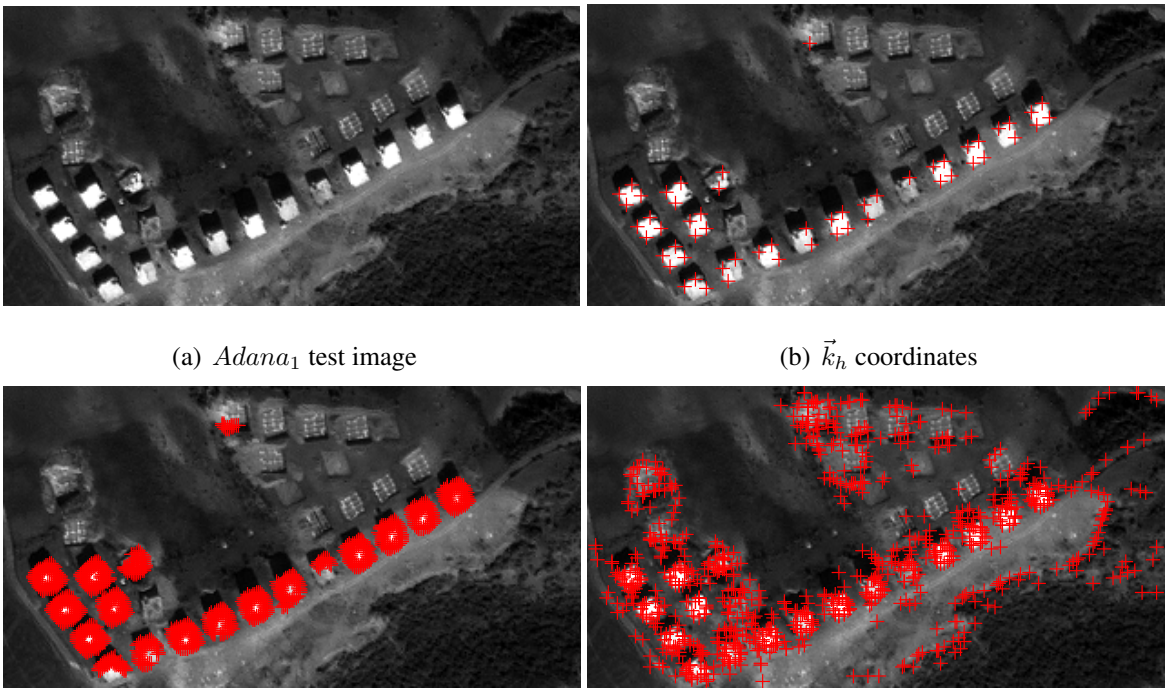
We pick the $Adana_1$ test image given in Fig. 3.3 and provide the spatial coordinates of local feature vectors extracted by three methods mentioned above. As can be seen, Harris corner detector gives only building corners. Although most of these are detected reliably, some of them are missing. Therefore, we have the least number of local feature vectors with the Harris corner detection method. The GMSR based method gives both corners and building edges. The Gabor filtering based method has similar results. However, some extra road segments and tree structures (resembling buildings) are also detected. Next, we use these local feature vectors to detect buildings.

3.3.2. Building Detection

We propose a probabilistic framework to detect buildings in this section. To explain our method in detail, we first explore nonparametric kernel based density estimation. Then, we focus on nonparametric variable kernel based density estimation. Next, we introduce our probabilistic building detection framework using it. Finally, we introduce data and decision fusion methods based on our probabilistic framework to detect buildings.

3.3.2.1. Kernel based Density Estimation

Each local feature vector indicates a building to be detected in the image. However, only one of them is not sufficient alone to detect a building. In fact, the more local feature vectors a building has, the more probable its detection. On the other hand, we do not know how many buildings are there in the image. Therefore, we formulate our building detection method with a probabilistic framework. To do so, we represent possible building locations as discrete joint



(a) $Adana_1$ test image (b) $\vec{k}_h$ coordinates

(c) $\vec{k}_g$ coordinates (d) $\vec{k}_f$ coordinates

Figure 3.3. The $Adana_1$ test image and local feature vector coordinates extracted with different methods

random variables. We estimate their probability density function (pdf) by taking local feature vectors as observations.

Since we do not know how many buildings are there in a given image, we benefit from nonparametric kernel density estimation. In this method, a kernel function is selected. Based on the observations of the random variable, the corresponding pdf is estimated. Silverman [76] defines the kernel density estimator for a discrete and bivariate pdf as follows. First, the bivariate kernel function, $K(x, y)$ should satisfy the conditions

$$\sum_{x}\sum_{y} K(x, y) = 1 \tag{3.16}$$

and

$$K(x, y) \geq 0 \ \ \forall (x, y) \tag{3.17}$$

Usually, $K(x, y)$ is taken as a symmetric pdf, the Gaussian function for instance. Then, the pdf

estimator with kernel $K(x, y)$ is defined by

$$p(x, y) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}, \frac{y - y_i}{h}\right) \tag{3.18}$$

where h is the window width (also called the smoothing parameter) and (x_i, y_i) for $i = 1, ..., n$ are observations.

If observations can not be represented reliably by a fixed kernel function, then a variable kernel function can be used. Therefore, the variable kernel based density estimation method allows adaptation of the amount of smoothing to the local density of the data (observation). Hence, the scale parameter is allowed to vary from one observation point to another. Besides, the estimate is constructed similarly to the classical kernel estimate. The pdf estimate given in Eqn. 3.18 then becomes

$$p_v(x, y) = \frac{1}{nh} \sum_{i=1}^{n} \frac{1}{\sigma_i} K\left(\frac{x - x_i}{h\sigma_i}, \frac{y - y_i}{h\sigma_i}\right) \tag{3.19}$$

where σ_i is the variable scale parameter for $i = 1, ..., n$.

3.3.2.2. Detecting Buildings using Variable Kernel based Density Estimation

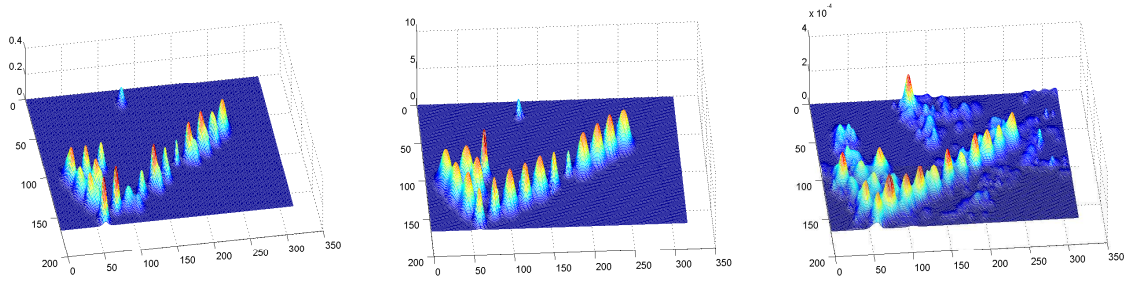
We use the variable kernel based density estimation method to detect buildings in a given image. As we mentioned previously, we use local feature vectors ($\vec{k}_h$, $\vec{k}_g$, $\vec{k}_f$) as observations to estimate the pdf. Without loss of generality, we explain pdf estimation on a generic local feature vector $\vec{k} = (x_i, y_i, \theta_i, w_i)$ for $i = 1, ..., K_i$. These vectors provide information on buildings to be detected. However, their spatial coordinates are not sufficient enough since they either represent building corners or edges. To detect a building, we need an ensemble of edges or corners. To achieve this, we adjust the effect of local feature vectors with respect to their orientation and weight. In doing so, we observed that for bright and dark building corners the gradient directions are towards the building center. For bright building edges, gradient directions are also towards building centers. Therefore, each local feature vector will have its effect as $\hat{x}_i = x_i + w_i \sin(-\theta_i)$ and $\hat{y}_i = y_i + w_i \cos(-\theta_i)$. In other saying, each local feature vector is shifted in the reverse direction of θ_i. We apply a weight in shifting to approximately locate the building center. Using

these adjusted and updated observations, we form the estimated pdf as

$$p_b(x,y) = \frac{1}{R}\sum_{i=1}^{K_i} \frac{1}{\sqrt{2\pi\sigma_i}} \exp\left(-\frac{(x-\hat{x}_i)^2 + (y-\hat{y}_i)^2}{2\sigma_i}\right) \tag{3.20}$$

where $\sigma_i = \beta w_i$. Here, $\beta = 0.2$ for $\vec{k}_h$, $\vec{k}_g$ and $\beta = 0.5$ for $\vec{k}_f$. We added β as a normalizing constant since we obtain weights by different methods. We will obtain modes of $p_b(x,y)$ in detecting buildings. Therefore, we did not use a normalized kernel in this equation. However, we normalized the final estimated pdf $p_b(x,y)$ by the normalizing constant R.

Before proceeding further, we provide the kernel density estimation results in Fig. 3.4 using three local feature vector extraction methods introduced earlier. As can be seen, for the Harris and GMSR based corner detector, the estimated pdf is smooth. For the Gabor filtering based method, the estimated pdf has more fluctuations in non building regions. The estimated



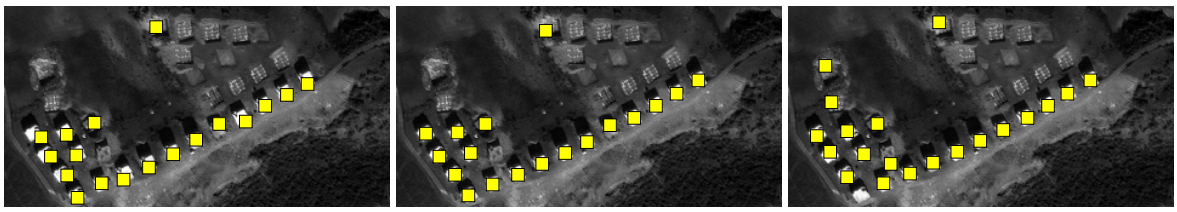
(a) $\vec{k}_h$ based pdf (b) $\vec{k}_g$ based pdf (c) $\vec{k}_f$ based pdf

Figure 3.4. The $Adana_1$ test image kernel density estimation results for three different local feature vector extraction methods

pdf $p_b(x,y)$ will be multimodal since we have unknown number of buildings to be detected in the image. We hypothesize that the modes of this pdf are possible building centers. Therefore, we detect building locations by the modes of $p_b(x,y)$. However, all modes do not correspond to a building center. Therefore, we assume that for a location to be a building center, it should have at least a minimum probability. We adjust this value in an adaptive manner as follows. Since we are detecting buildings in an urban area, we assume that there is at least one building there. Therefore, we pick the mode location having the highest probability as a building location, $(x_b, y_b) = \arg\max_{(x,y)} p_b(x,y)$. Then, we pick the remaining mode locations having probabilities at least $0.4 \times p_b(x_b, y_b)$ as building locations. By eliminating mode locations having probabilities less than $0.4 \times p_b(x_b, y_b)$, we eliminate false alarms. Here, we obtained 0.4

coefficient after extensive tests on our data set. It is also possible to choose this coefficient as equal to Otsu's automatically detected threshold value, however we obtained higher performance using 0.4 value. This method also automatically assigns probabilities to detected building locations. The higher probability the location has, the more probable it represents a building. This information may be of use in some other applications.

We provide the detected buildings in the $Adana_1$ test image in Fig. 3.5 by three local feature vector extraction methods mentioned above. As can be seen, for three methods almost all of the buildings are reliably detected. There is a missing building and a false alarm for the Harris corner and GMSR based local feature vector extraction methods. The false alarm rate increases to three with the Gabor filtering based local feature vector extraction method. We comment on the performance of these methods on several images in detail in the experiments section. Next, we introduce fusion methods to improve our building detection method.



(a) Harris buildings (b) GMSR buildings (c) Gabor buildings

Figure 3.5. Buildings detected by three different local feature vector extraction method from the $Adana_1$ test image

3.3.2.3. Data and Decision Fusion for Building Detection

The three local feature vector extraction methods extract different information from the same image. In Section 3.3.2.2, we separately used these to detect building locations. Their fusion may also improve our building detection results. Fortunately, the proposed probabilistic building detection method allows fusion of information. Therefore, in this section we introduce two fusion methods using our probabilistic framework to improve our building detection results.

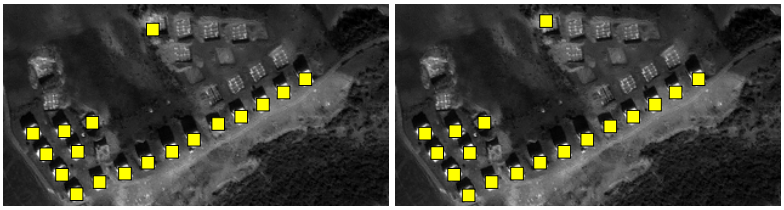
Our first method is based on data fusion. This method is straightforward, such that we use all the local feature vectors extracted with different methods as one unique group. In other saying, $\vec{k_F} = \{\vec{k}_h, \vec{k}_g, \vec{k}_f\}$. We estimate the pdf using this group. With the same method in the previous section, we detect buildings.

Our second method is based on decision fusion. Here, we mix the estimated pdfs by different methods and obtain a final pdf. While mixing the estimated pdfs, we assign a weight to each of them directly proportional to their maximum mode value. As we mentioned in the previous section, in detecting buildings from the estimated pdf we label the mode with the maximum value as a building. By normalizing three different pdfs this way, we can mix them and obtain the final pdf estimate as

$$p_D(x,y) = \frac{1}{R} \sum_{l=\{h,g,f\}} \frac{p_l(x,y)}{\max_{(x,y)} p_l(x,y)} \tag{3.21}$$

where $p_h(x,y)$, $p_g(x,y)$, $p_f(x,y)$, are the estimated pdfs from $\vec{k}_h$, $\vec{k}_g$, and $\vec{k}_f$. R is again the normalizing constant. We call this method as decision fusion, since we apply the fusion operation close to the building detection step. Again, we use the building detection method in the previous section on $p_D(x,y)$ to detect buildings.

We provide the detected buildings in the $Adana_1$ test image in Fig. 3.6 by the introduced data and decision fusion methods. As can be seen in this figure, for both data and decision fusion methods we have similar detection results with the Harris and GMSR based local feature extraction methods for this test image. In Section 3.7, we compare all building detection methods in detail.



(a) Data fusion (b) Decision fusion

Figure 3.6. Buildings detected by data and decision fusion methods from the $Adana_1$ test image

3.4. BUILDING DETECTION USING STEERABLE FILTERS

In this section, we propose another novel approach for building detection. For this purpose, we extract edges of these objects using a steerable filter set. Before using steerable filters, we apply bilateral filtering to eliminate noise terms. Then we apply steerable filters in different orientations. We take these filter responses as local features and detect buildings on these. We test

our algorithm on very high resolution panchromatic Ikonos satellite images and aerial images including buildings with diverse characteristics. Experimental results indicate practical usefulness of the algorithm.

3.4.1. Edge Detection with Steerable Filters

Edges and curvilinear structures are crucial features to detect buildings in remotely sensed images. For example, buildings generally have edges or curves around one center.

In order to extract object edges, herein we benefit from steerable filters. Steerable filters provide directional edge detection since they behave as band-pass filters in particular orientations. Differently from Gabor filters, steerable filters can be synthesized easily as a linear combination of a set of basis filters. In this study, we use steerable filter as shown by Freeman and Adelson [77].

For a symmetric Gaussian function $G(x,y) = exp(-(x^2+y^2))$, it is possible to define basis filters Gp_0 and $Gp_{\frac{\pi}{2}}$ as

$$Gp_0 = \frac{\partial}{\partial x}G(x,y) = -2x\exp(-(x^2+y^2)) \tag{3.22}$$

$$Gp_{\frac{\pi}{2}} = \frac{\partial}{\partial y}G(x,y) = -2y\exp(-(x^2+y^2)) \tag{3.23}$$

We can find a derivative in an arbitrary direction θ using the following rotation

$$Gp_\theta = cos(\theta)Gp_0 + sin(\theta)Gp_{\frac{\pi}{2}} \tag{3.24}$$

We represent shape of Gp_θ filter function for $\theta = 0$ filtering direction in spatial domain in Fig. 3.7. After obtaining steerable filter function in θ direction, we convolve the smoothed image $I_b(x,y)$ with filter Gp_θ ($J_\theta(x,y) = I_b(x,y) * Gp_\theta$), to detect edges in the θ direction. In $J_\theta(x,y)$, we expect high responses in edge locations perpendicular to the filtering direction. Therefore, we obtain local features by thresholding $J_\theta(x,y)$. To obtain an adaptive method, we

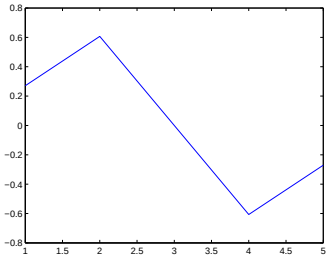


Figure 3.7. Steerable filter function (Gp_θ) in spatial domain for $\theta = 0$ filtering direction

pick the threshold value as the 20% of maximum magnitude in $J_\theta(x, y)$. We picked this value after extensive testing. After thresholding, we obtain a binary image with pixel locations having value one representing possible building features. As in the previous section, we apply connected components analysis to the thresholded image and obtain each local feature separately. We expect this novel local feature to behave more robust than our previous local features, since it also gives structural information.

We extract local feature vectors in all θ directions. Since we do not have a prior knowledge about the building alignments, these different directions are necessary. In this study, we pick our steerable filtering directions as $\theta \in [0, \pi/12, ..., 23\pi/12]$ interval. Therefore, we have a total of 12 filtering directions. After these operations, we may have either a straight line segment or an L shaped curve representing buildings in the image. This is due to the properties of the building shape and the steerable filtering operation. In fact, the L shaped corners are more valuable in representing building appearance. Therefore, we first classify our local features into two groups as straight line segments and curves.

To detect curved edges, Orrite *et al.* [78] developed a nice approach. They extended end points of curves, if extended end points of two curves intersect each other then they grouped these two curves as a close shape. In our problem, we have many buildings in a scene, hence applying this test to all edge couples may require too much computation time. Therefore, in this study we developed a novel and fast approach to detect curved edges. For feature classification, we apply the following test. First, we obtain the skeleton of the local feature [39]. Then, we detect the endpoints of this skeleton. Endpoints are chosen as two pixels which have only one neighbor pixel and which have highest Euclidean distance between each other. Assume that, we obtain two endpoints as (x_1, y_1) and (x_2, y_2). We demonstrate curved feature detection process in Fig. 3.8. We calculate (x_m, y_m) as the midpoint of the skeleton, and (x_o, y_o) as the midpoint

of the virtual line (dashed line) which connects endpoints. We obtain the Euclidean distance between (x_m, y_m) and (x_o, y_o). If the local feature is a straight line then (x_m, y_m) and (x_o, y_o) overlaps. Therefore, the distance between them equals to zero. On the other hand, if the local feature is a curve, then this distance is greater than zero. This way, we can classify local features as either a straight line or a curve.

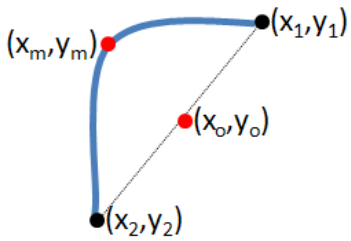
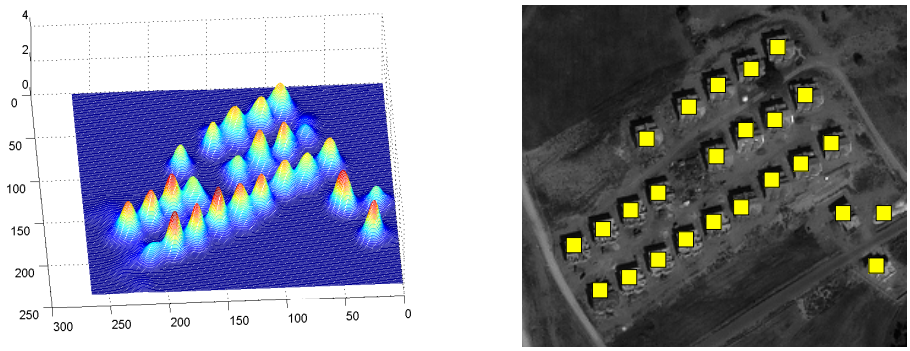


Figure 3.8. Curve detection example

3.4.2. Building Detection

After extracting and classifying steerable filtering features, we group edges using the probabilistic approach as in the previous section. We introduced kernel based density estimation method in Section 3.3.2.1. Here, we formed the pdf again using Eqn. 3.20. However, we choose σ_i as equal to 1 for curves, and 0.5 as straight lines. As a result, in final estimated $p_b(x, y)$ pdf curves have higher effect than straight lines. We provide kernel density estimation result in Fig. 3.9 at the left hand side. Comparing with Gabor feature based estimated pdf (in Fig. 3.4), steerable filtering based pdf has less fluctuations in non building regions. Comparing with Harris and GMSR based pdf's (in Fig. 3.4), steerable filtering based pdf is more successful to estimate dark buildings.

Using estimated pdf $p_b(x, y)$, we detect building locations using a similar method as in our previous approach. We hypothesize that the modes of pdf are possible building centers. Therefore, again we detect building locations by the modes of $p_b(x, y)$. In order to prevent false detections, we assume that for a location to be a building center, it should have at least a minimum probability. We pick the mode location having the highest probability as a building location, $(x_b, y_b) = \arg\max_{(x,y)} p_b(x, y)$. Then, we pick the remaining mode locations having probabilities at least $0.4 \times p_b(x_b, y_b)$ as building locations. As a result, we detect building locations automatically as in Fig. 3.9. As can be seen in this figure, we detected all buildings in the given scene. In Fig. 3.10 we provide a sample building detection result using steerable filters on



(a) p_b (b) buildings

Figure 3.9. Possible building centers for the $Adana_8$ test image and detected buildings

a large Ikonos image. However there are some false alarms especially on roads, overall building detection performance of proposed method is fairly good. We present experimental results of steerable filtering based method on our satellite and aerial image data set in Section 3.7.

Figure 3.10. Building detection results using steerable filters on a sample Ikonos image

3.5. BUILDING DETECTION USING COLOR INDICES

So far, we have used only grayscale information to detect urban region boundaries and buildings. If available, using color bands of captured image provides valuable information for object detection. Since our aerial images are captured in RGB color format, in this section we propose a novel approach to detect buildings using color information. For this purpose, we first extract areas of interest using invariant color features. We also extract shadow information using invariant color features. We use shadow segments to determine the illumination direction and verify building locations. Finally, we present a novel method to determine shapes of the buildings using the edge information.

3.5.1. Detecting Buildings

We provide a sample test image (with its buildings labeled manually) in Fig. 3.11. To detect buildings similar to the ones given in this image, we benefit from color invariants. We explore them next.

Figure 3.11. Manually labeled buildings in the $Sample_1$ aerial test image

3.5.1.1. Detecting Rooftop and Shadow Pixels

Color invariants help extracting color properties of objects without being affected by imaging conditions. Imaging conditions can be counted as, the illumination of the environment, the surface properties of the object, the highlights or shadows on the object, and the change of the angle of view. In literature, Gevers and Smeulders [79] proposed several color invariants. Ünsalan and Boyer [7] applied PCA to the data in order to decorrelate color components. By projecting the components to the uncorrelated random variables and calculating their slope, the normalized difference vegetation index is obtained. The color invariant used in this project is originated from this study. Here, we apply the same procedure to RGB images to obtain a new color invariant as

$$\psi_r = \frac{4}{\pi} \arctan\left(\frac{R-G}{R+G}\right) \tag{3.25}$$

where R stands for the red band and G stands for the green band of the color image. This color invariant has a value of unity for red colored objects independent of their intensity values. Similarly, it has a value of minus unity for green colored objects in the image. Therefore, the red rooftops (of buildings) can be easily segmented using ψ_r. Since most buildings have red rooftops in the test region, this invariant is of great use to detect buildings. To segment out the red rooftops

automatically, we benefit from Otsu's thresholding method.

Shadow information is dominantly found in the blue band of the RGB color space. So, we can propose a similar color invariant (to Eqn. 3.25) to enhance shadow regions as

$$\psi_b = \frac{4}{\pi} \arctan\left(\frac{B-G}{B+G}\right) \tag{3.26}$$

where B stands for the blue band of the color image. Again, in this invariant shadow regions have high intensity values. Therefore, they can easily be segmented out by a simple thresholding operation on the ψ_b image. To segment out the shadow regions automatically, we again benefit from Otsu's thresholding method.

3.5.1.2. Estimating the Illumination Direction

In literature, some of the researchers provided the illumination direction to the system manually [80]. In our study, we assume that the illumination direction can be estimated if a connected rooftop and shadow region couple can be obtained from the image. We consider the illumination direction as the direction of the line beginning from the center of the rooftop region to the center of the shadow region.

For the rooftop and shadow couple, if center of the rooftop region is at (x_b, y_b), and the center of the shadow region is at (x_s, y_s), then the illumination angle θ is

$$\theta = \arctan\left(\frac{|y_b - y_s|}{|x_b - x_s|}\right) \tag{3.27}$$

The quadrant θ lies is also important. We can adjust θ according to its actual quadrant as

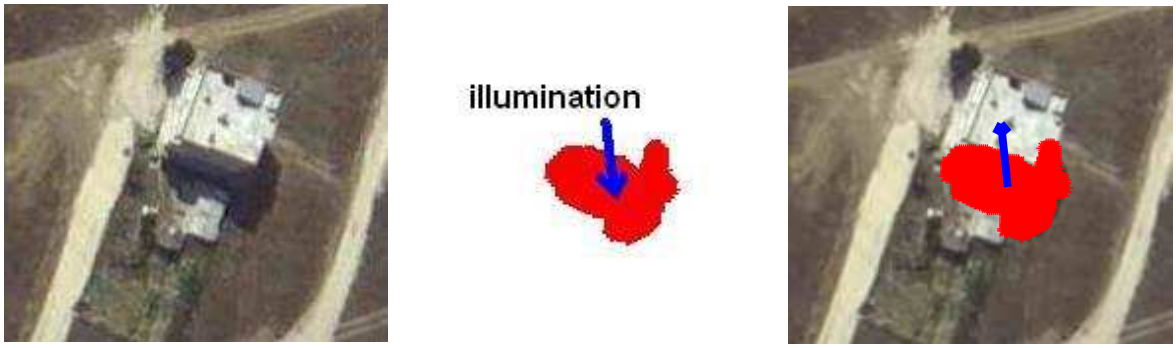
$$\theta = \begin{cases} \theta & \text{if} \quad x_s > x_b, y_s < y_b \\ \pi - \theta & \text{if} \quad x_s < x_b, y_s < y_b \\ \pi + \theta & \text{if} \quad x_s < x_b, y_s > y_b \\ 2\pi - \theta & \text{if} \quad x_s > x_b, y_s > y_b \end{cases} \tag{3.28}$$

3.5.1.3. Verifying the Building Appearance

If the rooftop (of a building) to be detected is not red, then our color invariant ψ_r may not be sufficient to detect it from the aerial image. To detect such rooftops (hence buildings), we have to look for other cues. Since we determined the illumination angle θ in Eqn. 3.28, this information may be of help to verify red rooftops as well as infer non-red rooftops. To do so, we introduce a hypothesis test such that; if we detect a shadow somewhere in the image it should originate from a building. Therefore, we check for the possible locations of a building based on the illumination direction and the center position of the shadow region as

$$(x_e, y_e) = (x_s + d\cos\theta, y_s + d\sin\theta) \tag{3.29}$$

where (x_e, y_e) represents the coordinates of the estimated building center. d is the possible distance that a building can be located. In this study, we use this distance as 17 pixels considering the size of the buildings. We provide a simple illustrative example for building verification in Fig. 3.12. In Fig. 3.12, the first image shows a sample building without a red rooftop. In the



(a) A non-red rooftop building (b) The detected shadow segment (c) Possible location of the building and illumination direction

Figure 3.12. Using the shadow information to detect non-red rooftop buildings

second image, dark region represents the shadow segment that is extracted using the color invariant ψ_b. Since the building rooftop is not red, the building location could not be detected by thresholding ψ_r. So, we use the illumination direction information to estimate the possible building location. The illumination angle and direction is calculated using the red rooftop and shadow couples of other buildings (in the image) as we have introduced in the previous section. We provide this result in the next image. We assume that, the building should be in the opposite direction of the illumination vector. We illustrate building center estimation process in the third image. We locate a 30×30 window on this center coordinates. Hereafter, we will call this region

as the estimated building segment. Size of window is chosen considering approximate sizes of buildings in test images. We will try to estimate this window size in our future studies.

3.5.2. Determining the Building Shape with a Novel Approach

Most buildings have rectangular shapes. To use this property, we introduce a novel box-fitting approach to edges of buildings. We use the Canny edge detector for extracting edges of candidate building regions [81]. This method first finds gradients the image using derivative of the Gaussian in both the vertical and horizontal directions. Gradients higher than a threshold are chosen as Canny edges. To define smoothing parameter of Gaussian kernel and the threshold value, we use default parameters of Canny edge detector function in Matlab. There, smoothing parameter of Gaussian kernel is chosen as 1, and the threshold value is chosen as 0.4 times of Otsu's automatically detected threshold value. From these Canny edges, the rectangle that represents the building will be constructed. The proposed algorithm works on both the red and non-red rooftop building segments. Therefore, our method does not specifically depend on the red rooftop information.

We introduce the building shape determination process for two purposes. First, presenting a reliable shape of a building is important for constructing a land map. Second, we verify appearance of buildings in estimated building segments (which are obtained by using only shadow and illumination information).

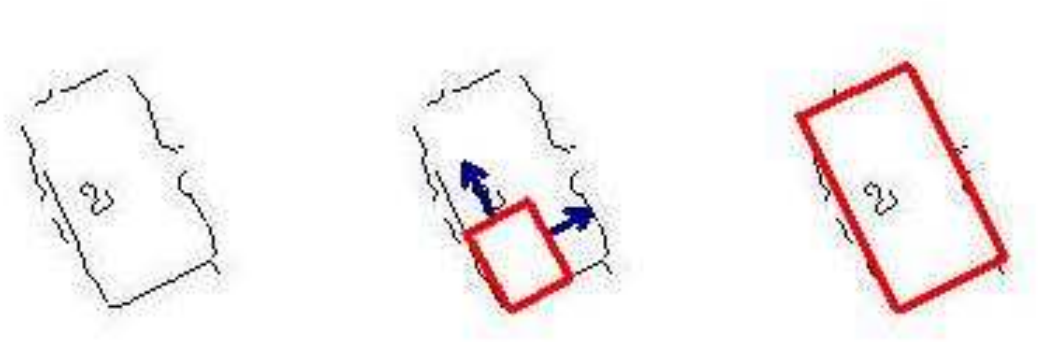
The proposed box-fitting method discards edges out of the building segments and processes edges of one segment at a time. Therefore, dealing with only candidate building edges decreases the number of unnecessary edges. For each segment, we detect possible corners of the line segments. We calculate the angle between lines for each corner. We call this angle as β. We choose the corner which has the smallest $|\pi/2 - \beta|$ value and satisfies $|\pi/2 - \beta| < \beta_{err}$ condition as a starting point. Due to the noise and the angle of the aerial camera, corners may not be exactly $\pi/2$ radians apart. We use a tolerance $\beta_{err} = 0,05\pi$ radians in our experiments to compensate this. We put the initial seed box on the corner which is chosen as the starting point. Then, the edges belonging to the box corner at the opposite side of the start point are swept outwards. We stop the iteration process when the energy E of the box shape reaches a minimum. Energy of the box shape is defined as the sum of minimum distances between building edge pixels and the box

edge pixels as

$$E = \sum_{i=1}^{n} min(\sqrt{(x_v(i) - x_e(j))^2 - (y_v(i) - y_e(j))^2}) \tag{3.30}$$

where, $(x_v(i), y_v(i))$ represents coordinates of ith pixel on the edges of the box shape. $(x_e(j), y_e(j))$ represents the jth pixel on the building edges.

Since the buildings we are looking for are constructed by rectangular shaped structures, it makes sense to try to fit rectangular shapes on the detected edges. The algorithm gives successful results even the edges are not well-determined, there is not a closed shape, or the corners are not found. The algorithm also needs less computation time comparing with the active contour methods. We provide an example to this box fitting method in Fig. 3.13. In Fig. 3.13, the first



(a) The edges of the building (b) Initial box position (c) The fitted box

Figure 3.13. Box fitting method to detect shape of the buildings

image represents edges that are found in a sample building segment. In the second and third images, we demonstrate the box-fitting approach. As can be seen in these image, an initial seed box is located on the corner which has smaller $|\pi/2 - \beta|$ angle between its edges. Then we sweep edges of this virtual box outwards until the smallest box energy (E) is achieved. In the third image, final building shape is presented.

We provide a real example to this box fitting method continuing from the non-red rooftop building given in Fig. 3.12. We provide the final extracted shape of this building in Fig. 3.14. As one outcome of this algorithm, we reject the building appearance if the box can not converge a shape (if energy can not be minimized) in this region. Therefore, we also verify the appearance

Figure 3.14. An example of the box fitting method to detect shape of the building

of buildings in the estimated building segments using this algorithm. To note here, using box fitting approach our purpose is not to detect exact building shapes. Study region can contain buildings with very different structures. However, by detecting approximate shapes we verify building appearance and obtain more planned land map using box fitting approach. We provide a sample result of box fitting method on an L-shaped building in Fig. 3.15.

Figure 3.15. A sample result of box fitting method on L-shaped building

3.5.3. Validity of Results

Our method gives reliable results for aerial images having buildings with red and non-red rooftop buildings. If there are non-red rooftops in the test image, then at least one red rooftop and shadow couple should be found in the image to determine the illumination angle. After determining the illumination angle, locations of buildings that have rooftop in different color can be estimated by our method.

3.6. DAMAGED BUILDING DETECTION USING SHADOW INFORMATION

Natural disasters such as earthquakes or hurricanes may cause a great damage to a region. Although these disasters are inevitable, it is still possible to minimize the problems afterwards.

After an earthquake or a hurricane, the road network may be damaged. Therefore, the region may not be accessible using ground transportation. It is also highly possible that the communication network to be damaged. These deficiencies may limit the information flow from the disaster region. However, it is utmost important for rescue planners to get reliable information from these regions to effectively guide their resources. To get reliable information from a disaster region, one possible solution is sending an aerial surveillance system. This system may collect aerial images from the disaster region. Although the images may be of use for rescue planners, it is still hard to manually locate damaged buildings in these images. With the same reasoning, automatically locating the damaged buildings after a military air strike is utmost importance to military personnel. This information may give insight on the success of the air strike. Therefore, automatic damaged building detection from aerial or satellite images is an important problem in remote sensing.

In this section, we present a novel approach for automatic detection of damaged buildings in color aerial images. Our method is based on color invariants for building rooftop segmentation as in Section 3.5. Then, here we benefit from grayscale histogram to extract shadow segments. After building verification using shadow information, we define a new damage measure for each building. Experimentally, we show that using our damage measure it is possible to discriminate nearby damaged and undamaged buildings on aerial images.

3.6.1. Detecting Buildings and their Shadows

In our study, we assume that the damaged region is not imaged beforehand. Therefore, it is not possible to compare two images (as in standard change detection algorithms) to detect damaged buildings. We have to detect the damaged buildings using just one image (obtained after the earthquake or the air strike). In order to extract the building rooftops we benefit from invariant color features. In Section 3.5, we were able to locate the building rooftops in a reliable manner on color aerial images. Here also our first aim is to detect buildings and their shadows automatically as in Section 3.5. Then, we will use this information to define a measure to estimate the degree of damage. We provide a sample test image in Fig. 3.16 to explain our method. As can be seen in Fig. 3.16, there are only undamaged buildings in this region. We benefit from color invariants and grayscale information to extract rooftop and shadow segments from this test image. We explore them in detail next.

Figure 3.16. $Damage_1$ test image from our aerial image dataset

In Section 3.5, we used ψ_b color invariant feature to detect shadow segments. In extensive tests, we observed that using only ψ_b color invariant feature can lead to false alarms since it also detects regions which are more or less in blue color. Therefore, here we used grayscale information to detect shadows. In our experiments we observed that using grayscale histogram of image can also provide information for robust shadow detection. To extract shadow segments from grayscale histogram of the image, we first smooth the histogram with a median filter. Since, shadows generally appear in the darker regions of an image, we choose the first local minimum in the histogram as the threshold value. We extract shadow segments by thresholding the grayscale image with this automatically calculated threshold value. Using this method, we obtain the rooftops and shadow segments as in Fig. 3.17. In Fig. 3.17, blue segments represent detected

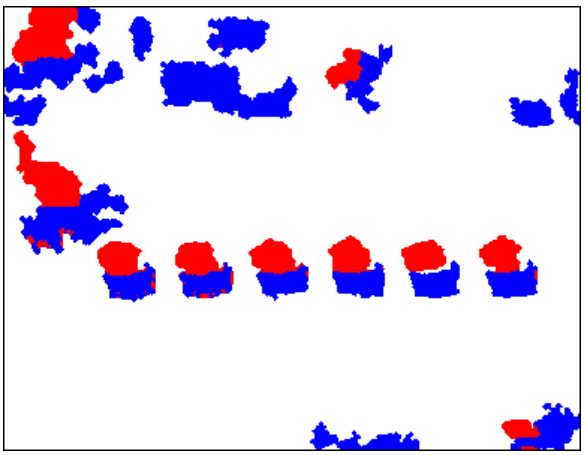


Figure 3.17. Building rooftop and shadow segments in $Damage_1$ image

shadows and red segments represent detected red rooftops in the $Damage_1$ test image. We provide the detection results on a blank image to increase visibility of segments.

As in the previous section, we assume that the illumination direction can be estimated if a connected rooftop and shadow region couple can be located in the image. Here, we use the same approach to detect centers of red and non-red building rooftops. We assume (x_b, y_b) coordinates as detected building centers.

3.6.2. Measuring the Degree of Damage

As we obtain the rooftop and shadow segments, we define a measure to determine the degree of damage. For this purpose, we calculate the ratio of rooftop and shadow areas for each building as

$$r = \frac{N}{M} \tag{3.31}$$

where N is the area of the rooftop segment and M is the area of the corresponding shadow segment. Since shadow and rooftop areas are larger for undamaged buildings, this ratio gives similar results. But if the building is decayed or if there is a structural damage on it, its shadow region will be smaller which leads rooftop to shadow ratio to have higher values. Again, to note here, we do not have image of the test region taken beforehand. Therefore, this ratio gives important information about the degree of damage using a single image.

3.7. EXPERIMENTAL RESULTS

In this section, we present experimental results of our building detection methods. We test our methods on panchromatic Ikonos images and aerial images. We also use aerial images in RGB color format to test our color invariant feature based algorithms.

In these images, the size and shape of buildings, their proximity, environment, and contrast of the building rooftops with respect to background all differ. These test images are specifically selected to represent wide and diverse building and region characteristics. In the following subsections, we analyze detection results of proposed methods quantitatively. At the end of the section, we compare performances, advantages and disadvantages of each building detection method. In the following subsection, we start with analyzing building detection result of SIFT and graph theory based algorithm.

3.7.1. Building Detection Using SIFT Descriptors and Graph Theory

Having detected urban areas in Section 2.5.1, here we concentrate on detecting buildings. As can be seen in the second columns of Figs. 3.18, 3.19, 3.20, and 3.21, our method detects buildings in each urban area fairly well. To quantify building detection results, we apply the following methodology. If a part of a building is detected, we assume it to be detected correctly. If our method detects a building multiple times (especially for large buildings), we assume it to be detected correctly. If a building is in construction (showing building characteristics), we expect our method to detect it. Based on these assumptions, we provide the building detection performances for all test images in Table 3.1. In this table, TP stands for the total number of buildings correctly detected in the image. FA stands for the total number of false alarms in the image. We also tabulate the total number of buildings in each test image.

From a total of 850 buildings in 30 different panchromatic satellite images, our method detected 751 of them correctly. This corresponds to a $\mathbf{TP} = \mathbf{88.4}\%$. There are 140 false alarms in building detection. This corresponds to a $\mathbf{FA} = \mathbf{16.5}\%$. On such a diverse test set, these results are very promising. Next, we consider interesting test images in detail. The lowest building detection rate is obtained on the $Adana_{18}$ image. The main reason for this poor performance is that, buildings are closely spaced and they are small in this test site. Therefore, some of the buildings could not be detected. The highest false positive rate is obtained on the $Adana_{21}$ image. The reason for this poor performance is viewing angle of the satellite. The highest building detection rate is obtained on the $Adana_{2,4,7}$, $Ankara_{1,3,4}$, and $Istanbul_{1,3,4}$ images. In all these test images, buildings are well separated and distinctive. Therefore, our method works fairly well. One of the lowest false alarm rate is obtained on the $Adana_{14}$ image. In this test image, there are no building like structures besides the actual buildings. This led to a low false alarm rate.

Beside these test images, there are other noteworthy test sites. For the $Adana_5$ test image, buildings are closely spaced. However, most of the buildings are correctly detected. For the $Adana_{11}$ test image, buildings are occluded by trees. It is even hard for a human expert to detect buildings in this image. However, again most of the buildings are correctly detected. For the $Adana_{15}$ test image, there are regularly spaced trees near buildings. One may think that, our building detection method may fail. However, for this test image there is only one false alarm and most of the buildings are correctly detected. In the $Adana_8$ and $Adana_9$ test images, the contrast

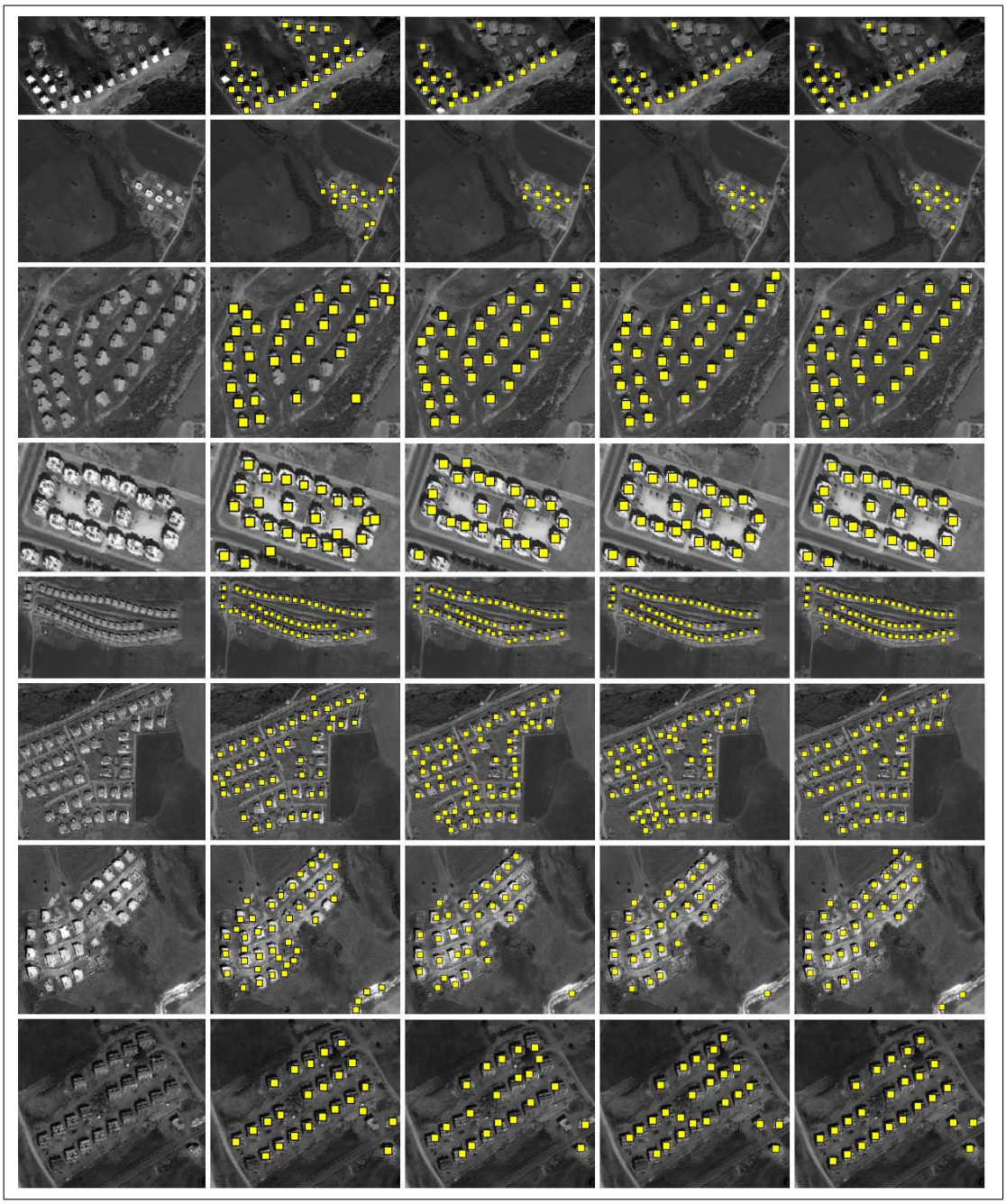


Figure 3.18. Building detection test results for *Adana* images (1 to 8) for each row separately. First column: original test images; second column: detected buildings using SIFT based algorithm; third column: detected buildings using Gabor based algorithm; fourth column: detected buildings by fusing features; fifth column: detected buildings with steerable filters

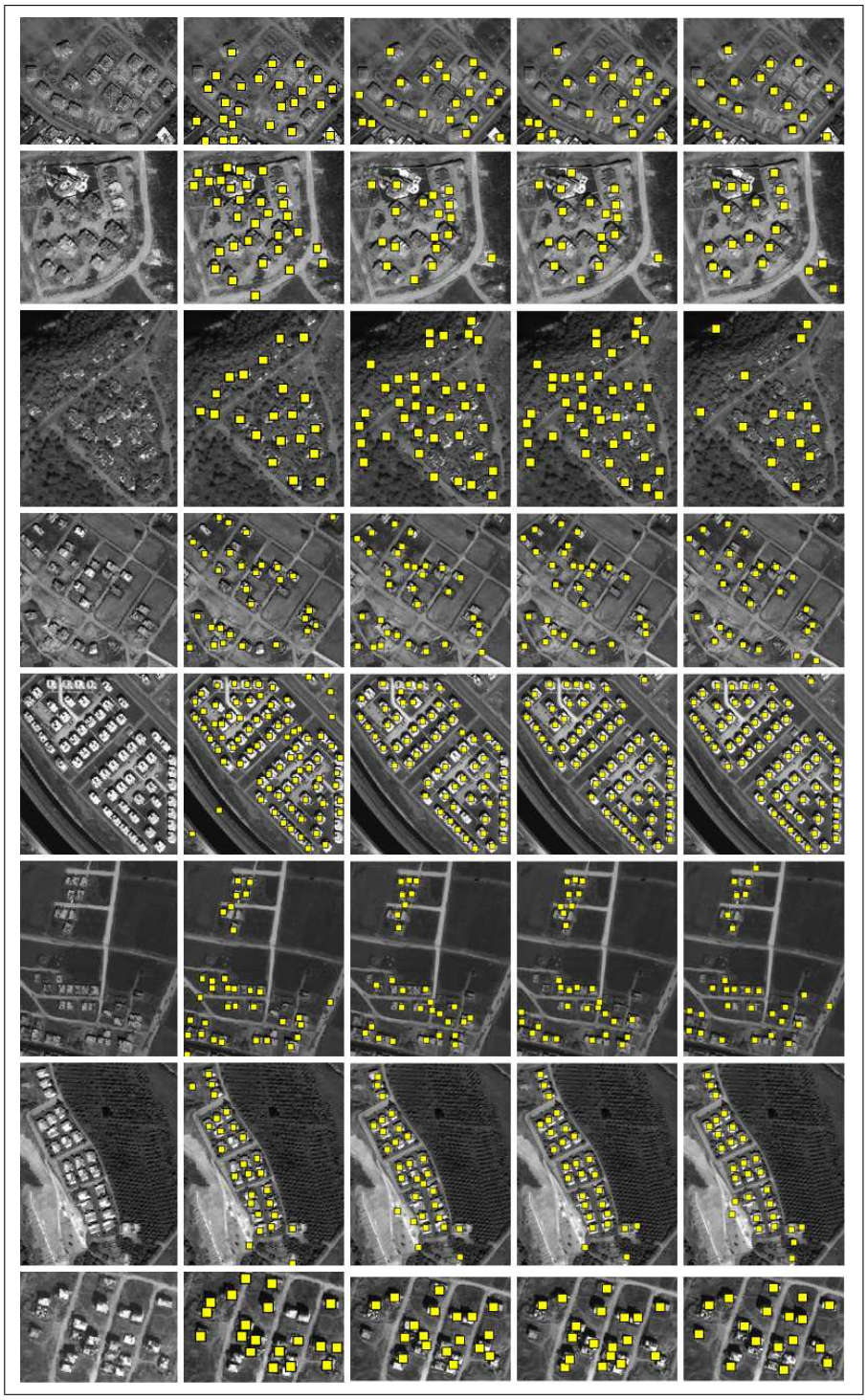


Figure 3.19. Building detection test results for *Adana* images (9 to 16) for each row separately. First column: original test images; second column: detected buildings using SIFT based algorithm; third column: detected buildings using Gabor based algorithm; fourth column: detected buildings by fusing features; fifth column: detected buildings with steerable filters

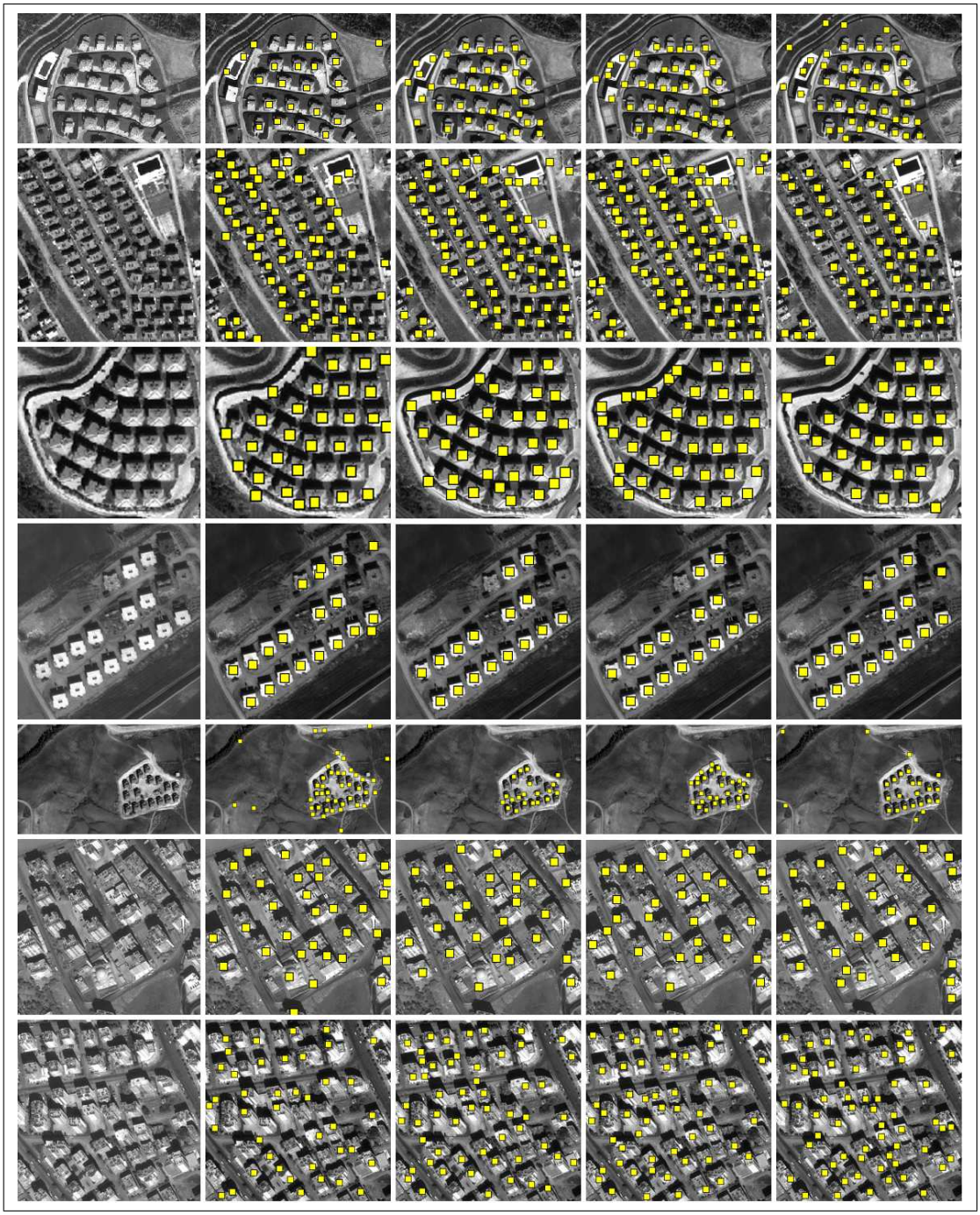


Figure 3.20. Building detection test results for *Adana* images (17 to 23) for each row separately. First column: original test images; second column: detected buildings using SIFT based algorithm; third column: detected buildings using Gabor based algorithm; fourth column: detected buildings by fusing features; fifth column: detected buildings with steerable filters

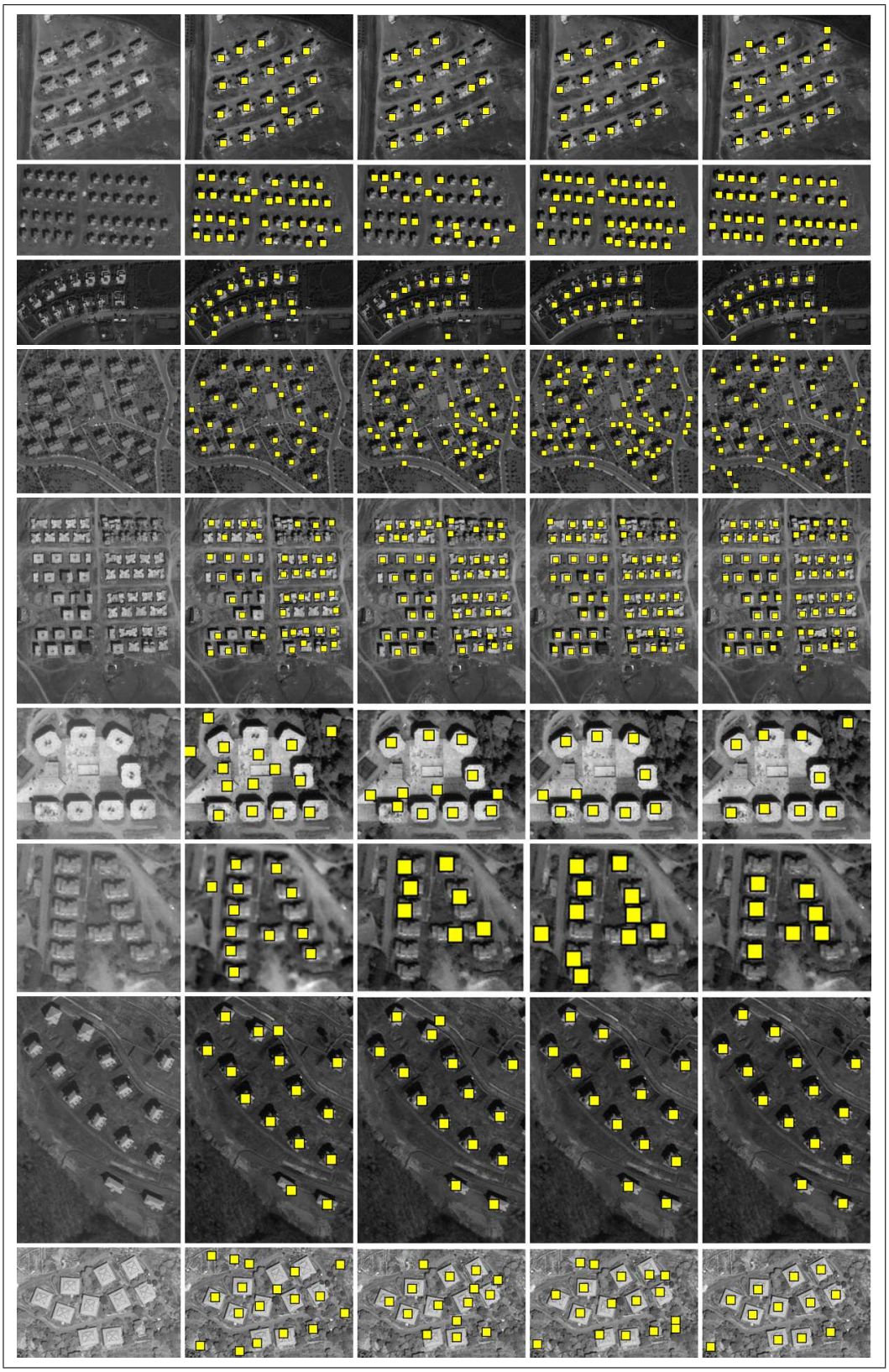


Figure 3.21. Building detection test results for $Ankara_{1to5}$ and $Istanbul$ images (1 to 4) for each row separately. First column: original test images; second column: detected buildings using SIFT based algorithm; third column: detected buildings using Gabor based algorithm; fourth column: detected buildings by fusing features; fifth column: detected buildings with steerable filters

Table 3.1. Building detection performances of SIFT based approach for Ikonos test images.

Image Name	**Buildings**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Adana_1$	29	25	4	86.2	13.8
$Adana_2$	9	9	6	100.0	66.7
$Adana_3$	31	27	1	87.1	3.2
$Adana_4$	21	21	4	100.0	19.0
$Adana_5$	54	49	1	90.7	1.9
$Adana_6$	47	45	6	95.7	12.8
$Adana_7$	28	28	9	100.0	32.1
$Adana_8$	24	21	2	87.5	8.3
$Adana_9$	24	23	5	95.8	20.8
$Adana_{10}$	14	13	9	92.9	64.3
$Adana_{11}$	21	19	5	90.5	23.8
$Adana_{12}$	32	29	4	90.6	12.5
$Adana_{13}$	67	60	2	89.6	3.0
$Adana_{14}$	33	29	0	87.9	0.0
$Adana_{15}$	28	27	6	96.4	21.4
$Adana_{16}$	23	17	1	73.9	4.3
$Adana_{17}$	24	21	8	87.5	33.3
$Adana_{18}$	70	48	13	68.6	18.6
$Adana_{19}$	24	23	7	95.8	29.2
$Adana_{20}$	20	17	2	85.0	10.0
$Adana_{21}$	18	16	21	88.9	116.7
$Ankara_1$	18	18	1	100.0	5.6
$Ankara_2$	44	34	1	77.3	2.3
$Ankara_3$	14	14	5	100.0	35.7
$Ankara_4$	23	23	5	100.0	21.7
$Ankara_5$	61	47	0	77.0	0.0
$Istanbul_1$	11	11	4	100.0	36.4
$Istanbul_2$	13	12	0	92.3	0.0
$Istanbul_3$	14	14	1	100.0	7.1
$Istanbul_4$	11	11	7	100.0	63.6
Total	**850**	**751**	**140**	**88.4**	**16.5**

between buildings and background is fairly low. However, correct building detection rates for these images are again reasonable. Based on these experimental results, we can conclude that our building detection method works fairly well on a diverse test set. There are minor shortcomings. First, very closely spaced buildings can not be separated. They are detected as a single building. Second, the contrast between the background and the building is very important for detection.

3.7.1.1. Tests on Different Modules

In Table 2.2, we tabulated the effect of different parameters on urban region detection by our SIFT based method. Here, we also provide the building detection results based on the same variations in Table 3.2.

Table 3.2. Building detection results on the $Adana_8$ image, parameter variations.

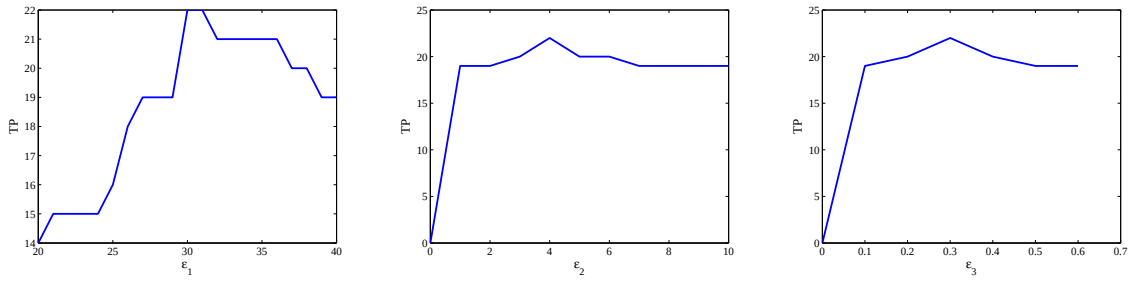
	Normal BF		**Fast BF**	
Template	TP	FA	TP	FA
Bright	17	1	17	1
Dark	21	2	19	2
Both	22	2	20	2

As can be seen in Table 3.2, in detecting buildings the bright building template again has the lowest performance. Using the dark building template or both templates improves the building detection performance. Unlike the urban area detection case, the type of the bilateral filter affects the building detection performance. Using the fast bilateral filter implementation slightly decreases the building detection performance in all template settings.

3.7.1.2. Tests on Parameter Values

In order to validate our parameter settings, we explore them in detail here. As a benchmark, we pick the building detection results (in terms of TP) on the $Adana_8$ test image. We change the value of each parameter and plot the TP values in Fig. 3.22. As can be seen, for ϵ_1 the optimal parameter is around 30. As we increase ϵ_1, the TP performance does not decrease drastically. For ϵ_3, the acceptable value is around 4. Increasing ϵ_3 further does not change the result further. We obtain a similar result for ϵ_4. The optimal value is around 0.1 and increasing ϵ_4 does not affect the result much. These experiments indicate the fairly robust characteristics of our parameter adjustment methods. Further details on the physical meanings of these parameters

can be found in previous sections.

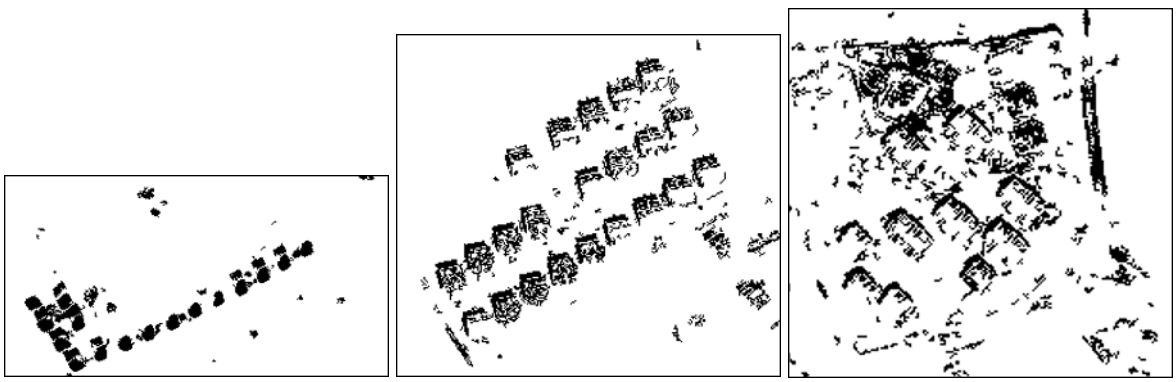


(a) TP vs. ϵ_1 (b) TP vs. ϵ_3 (c) TP vs. ϵ_4

Figure 3.22. TP vs. ϵ values for the building detection performance on the $Adana_8$ test image

3.7.1.3. Comparison with Derivative of Morphological Profiles

We also compare our building detection method with the well-known derivative morphological profiles (DMP) method [4]. To note here, DMP is not introduced for building detection alone. However, because of its strength it can also be used for building detection in panchromatic images. Therefore, we pick three test images and provide their segmentation results using DMP in Fig. 3.23. In testing DMP, we applied its principal components analysis based implementation. In segmentation, we picked the best threshold value for the $Adana_8$ test image. As can be seen in Fig. 3.23, detected buildings using DMP are not as good as our method. The main reason for this difference is that, we designed our method to building and urban area detection alone. On the other hand, the time needed for DMP operation on the $Adana_8$ test image is 19.19 sec. This timing is much less than the time needed for our method. Next, we discuss our method's timing requirements in detail.



(a) $Adana_1$ (b) $Adana_8$ (c) $Adana_{10}$

Figure 3.23. DMP test results on the $Adana_1$, $Adana_8$, and $Adana_{10}$ images

3.7.1.4. Computation Times

We finally tabulate the time needed for building detection. To note here, timing directly depends on the test image. As the number of buildings in a test image increases, number of local features will also increase. Therefore, the graph matching and graph cut algorithms will need more computation times. To give an idea for the possible reader, we consider the $Adana_8$ test image as a benchmark. We tabulate all CPU timings for each module in Table 3.3. In reporting these results, we used a PC with Intel Core2Due processor with 2.13 GHz. clock speed and having 4 GB of RAM. We used Matlab as our coding platform.

Table 3.3. CPU times (in sec.) for SIFT based building detection the $Adana_8$ test image.

#	**Module / Template**	**Dark**	**Both**
I	Upsampling	0.19	0.19
II	Bilateral filtering	62.67	63.89
III	Bilateral filtering, fast	9.92	9.92
IV	SIFT features	0.28	0.38
V	Graph matching	12.86	36.18
VI	Graph cut	137.66	329.50

In Table 3.3, we provide both normal and fast bilateral filtering implementations. Similarly, we provide the computation times for using only the dark building template and both templates. We can summarize different scenarios as follows. Using normal bilateral filtering and both templates, urban area detection operation requires 100.64 sec. In the same setting, building detection requires 430.14 sec. This scenario is for obtaining the best performances for both urban area and building detection. If we can tolerate slightly lower detection performances, then we can use fast bilateral filtering and only the dark template. In this scenario, urban area detection requires only 23.25 sec. Here, building detection only requires 160.91 sec. The possible reader should select the suitable scenario (both in terms of detection performance and CPU time needed) for his or her needs.

3.7.2. Building Detection Using Harris, GMSR and Gabor based Local Features

We test our building detection algorithm based on three different feature vectors on 32 high resolution panchromatic Ikonos images taken from Istanbul, Ankara and Adana cities of Turkey. We also tested the algorithm on aerial images taken from Istanbul city. We first tabulate building

detection results using Gabor filtering based features. Then we consider fusion of the proposed three features.

3.7.2.1. Building Detection Results on Satellite Images

We provide Gabor filtering based building detection performance in Table 3.4. Using Gabor feature based algorithm, we detected 784 of 911 buildings correctly. Unfortunately, if false features are located closely, sum of their probabilities generated a local maximum point in voting matrix. We could not prevent detection of these false local maximums as buildings. As a result, we obtain TP as $\mathbf{86.1}\%$ and FA as $\mathbf{19.4}\%$ for 911 buildings. Considering diverse characteristics of buildings in our test image data set, these results are encouraging. We also provide the detected buildings for each test image in the third columns of Figs. 3.18, 3.19, 3.20, and 3.21.

3.7.2.2. Building Detection Results on Aerial Images

Our Gabor filtering based local features can also work on aerial images. We tabulate the detection results in Table 3.5. In our aerial image data set, using Gabor feature based algorithm we detected 575 of the 652 buildings correctly. Very high resolution of aerial images bring some problems. Visible small details on ground surface lead to false feature detection. Unfortunately, if false features are located closely, sum of their probabilities generated a local maximum point in voting matrix. Here also we could not prevent detection of these false local maximums as buildings. As a result, we obtain TP as $\mathbf{88.2}\%$ and FA as $\mathbf{66.1}\%$.

We also provide the building detection results in Figs. 3.24, 3.25, 3.26, and 3.27. In these figures, original grayscale aerial images are presented in the first column, and Gabor feature based detection results are presented in the second column.

3.7.3. Building Detection by Fusing Different Local Features

Here, we test our fusion of local features based method on panchromatic Ikonos satellite and aerial image data sets. Again, we provide building detection results for both aerial and Ikonos satellite images in a quantitative manner. Here we also report the performance of all three local feature vector extraction methods, data and decision fusion methods separately. The most important advantage of our novel building detection method is its computation time and robustness to false alarms. Therefore, we examine the computation time of each building detection module in

Table 3.4. The building detection performance of Gabor filtering based local features on satellite images.

Image Name	**Buildings**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Adana_1$	18	17	3	94.4	16.7
$Adana_2$	9	9	1	100.0	11.1
$Adana_3$	31	31	0	100.0	0.0
$Adana_4$	21	20	1	95.2	4.8
$Adana_5$	54	45	5	83.3	9.3
$Adana_6$	47	45	10	95.7	21.3
$Adana_7$	28	27	4	96.4	14.3
$Adana_8$	24	20	1	83.3	4.2
$Adana_9$	24	16	6	66.7	25.0
$Adana_{10}$	14	11	6	78.6	42.9
$Adana_{11}$	21	16	22	76.2	104.8
$Adana_{12}$	32	28	4	87.5	12.5
$Adana_{13}$	67	61	1	91.0	1.5
$Adana_{14}$	33	25	3	75.8	9.1
$Adana_{15}$	28	26	4	92.9	14.3
$Adana_{16}$	23	16	1	69.6	4.3
$Adana_{17}$	24	23	15	95.8	62.5
$Adana_{18}$	70	59	15	84.3	21.4
$Adana_{19}$	24	24	8	100.0	37.5
$Adana_{20}$	20	16	0	80.0	0.0
$Adana_{21}$	18	15	5	83.3	33.3
$Adana_{22}$	27	25	6	92.6	22.2
$Adana_{23}$	48	43	12	89.6	25.0
$Ankara_1$	18	17	0	94.4	0.0
$Ankara_2$	44	20	2	45.5	4.5
$Ankara_3$	14	12	1	85.7	7.1
$Ankara_4$	23	23	31	100.0	147.8
$Ankara_5$	61	54	1	88.5	1.6
$Istanbul_1$	8	8	4	100.0	50.0
$Istanbul_2$	13	7	0	53.8	0.0
$Istanbul_3$	14	14	1	100.0	7.1
$Istanbul_4$	11	11	4	100.0	36.4
Total	**911**	**784**	**177**	**86.1**	**19.4**

Table 3.5. The building detection Performance of the Gabor filtering based local features on aerial images.

Image Name	**Buildings**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Aerial_1$	17	17	3	100.0	17.64
$Aerial_2$	27	21	51	77.8	188.9
$Aerial_3$	6	6	15	100.0	250.0
$Aerial_4$	9	9	17	100.0	188.9
$Aerial_5$	11	10	13	90.9	118.2
$Aerial_6$	16	15	5	93.7	31.25
$Aerial_7$	9	8	16	88.9	177.8
$Aerial_8$	11	11	6	100.0	54.5
$Aerial_9$	42	33	17	78.6	40.5
$Aerial_{10}$	50	44	19	88.0	38.0
$Aerial_{11}$	47	41	25	87.2	53.2
$Aerial_{12}$	57	43	9	75.4	15.8
$Aerial_{13}$	30	27	35	90.0	116.7
$Aerial_{14}$	19	18	30	94.7	157.8
$Aerial_{15}$	57	49	37	85.9	64.9
$Aerial_{16}$	56	45	15	80.3	26.7
$Aerial_{17}$	44	40	36	90.9	81.8
$Aerial_{18}$	65	63	44	96.9	67.7
$Aerial_{19}$	11	10	9	90.9	81.8
$Aerial_{20}$	16	15	12	93.7	75.0
$Aerial_{21}$	52	50	17	96.1	32.7
Total	**652**	**575**	**431**	**88.2**	**66.1**

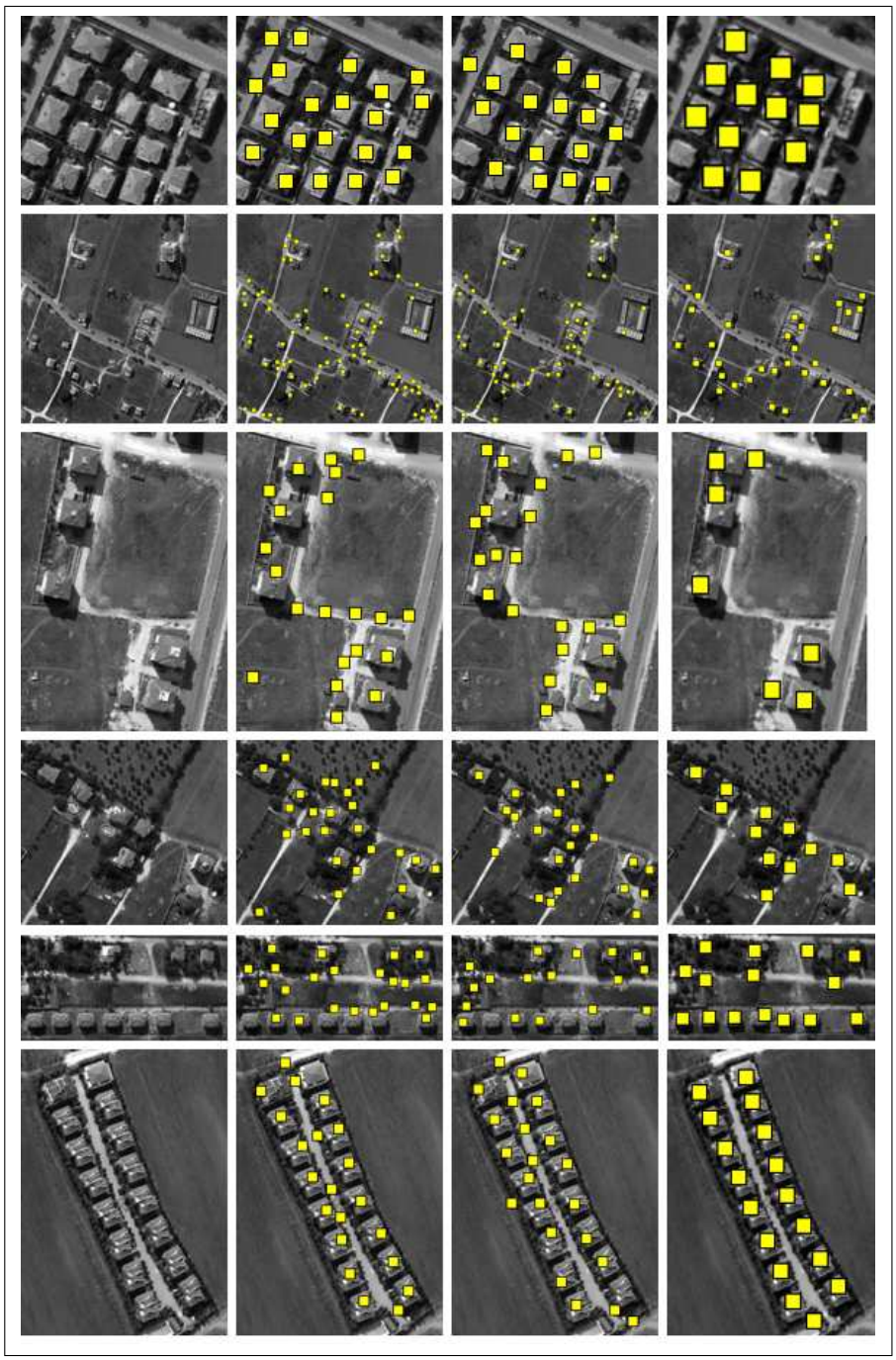


Figure 3.24. Building detection test results for $Aerial$ images (1 to 6) for each row separately. First column: original test images; second column: detected buildings using Gabor filtering based local features; third column: detected buildings by fusing features; fourth column: detected buildings by steerable filtering features

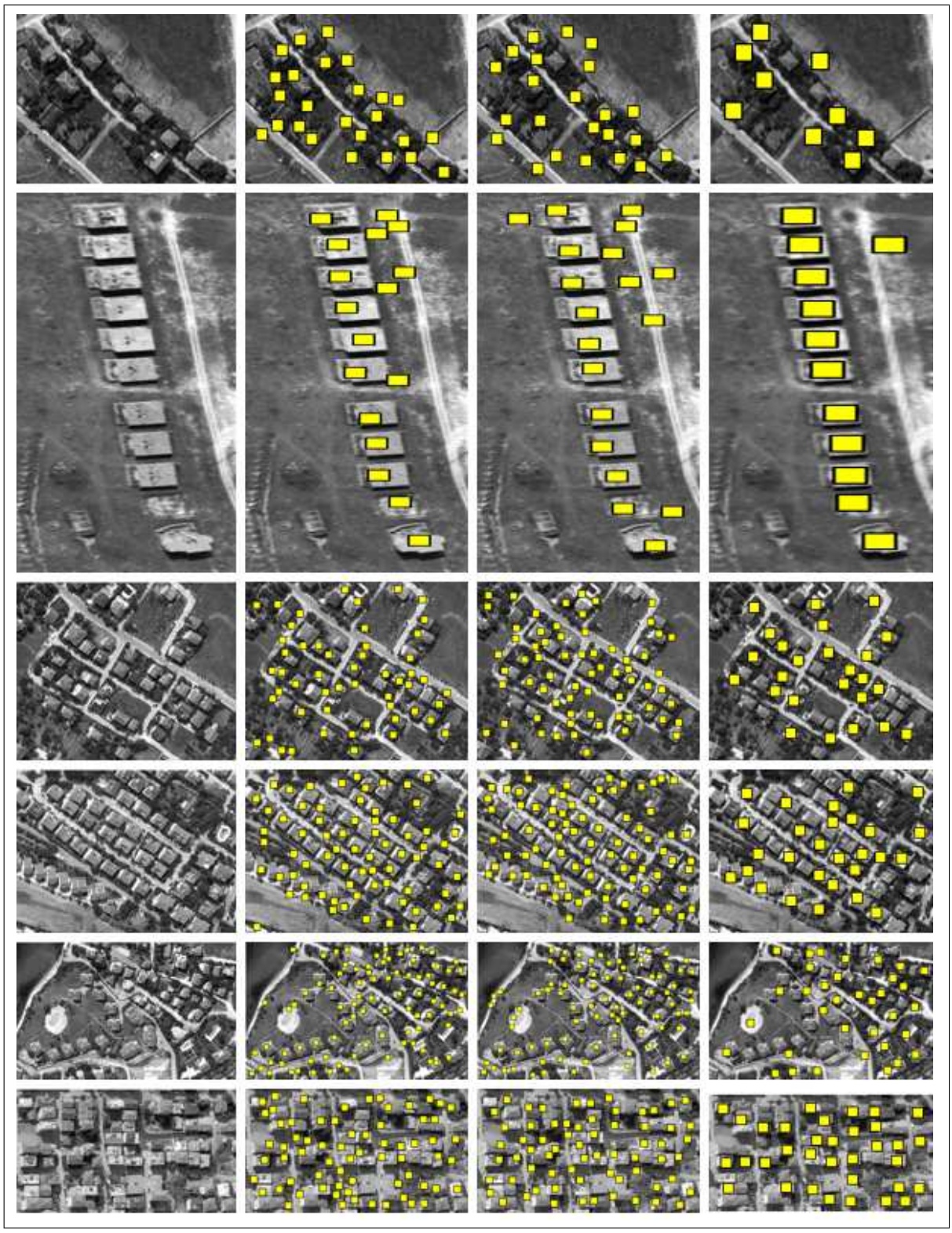


Figure 3.25. Building detection test results for *Aerial* images (7 to 12) for each row separately. First column: original test images; second column: detected buildings using Gabor filtering based local features; third column: detected buildings by fusing features; fourth column: detected buildings by steerable filtering features

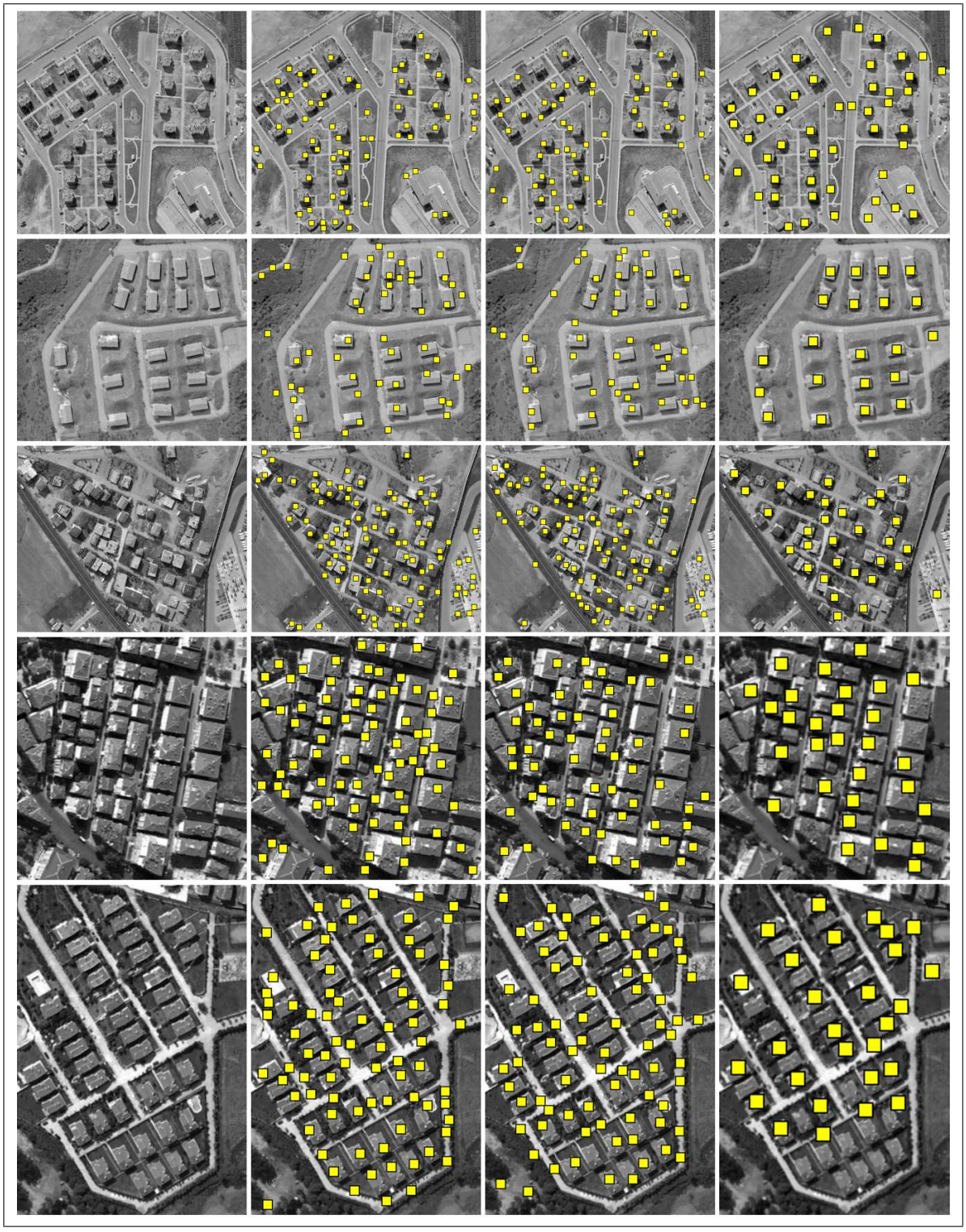


Figure 3.26. Building detection test results for *Aerial* images (13 to 17) for each row separately. First column: original test images; second column: detected buildings using Gabor filtering based local features; third column: detected buildings by fusing features; fourth column: detected buildings by steerable filtering features

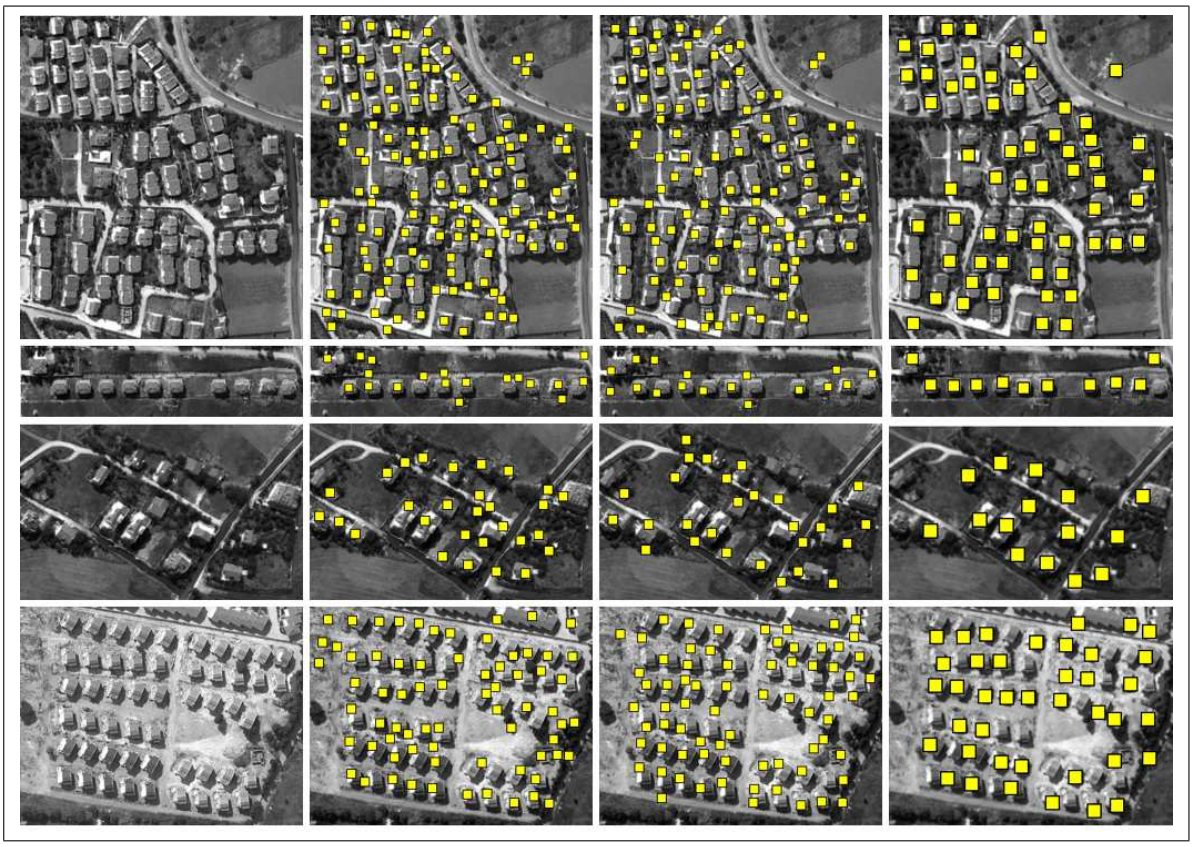


Figure 3.27. Building detection test results for $Aerial$ images (18 to 21) for each row separately. First column: original test images; second column: detected buildings using Gabor filtering based local features; third column: detected buildings by fusing features; fourth column: detected buildings by steerable filtering features

a separate section.

3.7.3.1. Building Detection Results on Satellite Images

We first test our building detection method on 32 Ikonos satellite images as in the previous section. To observe the performance of each local feature extraction and fusion method, we provide their building detection performances over all test images in Table 3.6.

Table 3.6. Building detection performances for Ikonos satellite images.

Method	**TP**	**FA**	**TP** $(\%)$	**FA** $(\%)$
Harris	699	186	76.7	20.4
GMSR	792	215	86.9	23.6
Gabor filtering	780	187	85.6	20.5
Data fusion	824	206	90.5	22.6
Decision fusion	**833**	**188**	**91.5**	**20.7**

As can be seen in Table 3.6, Harris corner based local feature vector extraction method has the lowest detection performance. GMSR and Gabor filtering based local feature vector extraction methods have similar detection performances. Both performances are also far better than the Harris corner based method. In both data and decision fusion methods the performance increases significantly. However, using decision fusion TP reaches $\mathbf{91.5}\%$ with $\mathbf{20.7}\%$ FA rate. This result is remarkable on such a diverse satellite image set.

We evaluate building detection performances for Ikonos satellite images in Table 3.7. As can be seen, 833 of the 911 buildings are detected correctly, which corresponds to 91.4% detection performance. There are only 188 false positives. We provide the detected buildings for each test image in the fourth columns of Figs. 3.18, 3.19, 3.20, and 3.21.

3.7.3.2. Building Detection Results on Aerial Images

We also test our probabilistic building detection method based on decision fusion on 21 aerial images. The total number of buildings in these images is 652. We evaluate building detection performances for our aerial image set in Table 3.8.

As can be seen in Table 3.8, we detected 560 of the 652 buildings. Unfortunately, we also

Table 3.7. Building detection performances using decision fusion on Ikonos satellite images.

Image Name	**Buildings**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Adana_1$	18	17	1	94.4	5.6
$Adana_2$	9	8	0	88.9	0.0
$Adana_3$	31	31	1	100.0	3.2
$Adana_4$	21	21	1	100.0	4.8
$Adana_5$	54	47	0	87.0	0.0
$Adana_6$	47	45	12	95.7	25.5
$Adana_7$	28	26	1	92.9	3.6
$Adana_8$	24	23	3	95.8	12.5
$Adana_9$	24	18	5	75.0	20.8
$Adana_{10}$	14	11	7	78.6	50.0
$Adana_{11}$	21	19	18	90.5	85.7
$Adana_{12}$	32	27	2	84.4	6.3
$Adana_{13}$	66	65	0	98.5	0.0
$Adana_{14}$	33	26	5	78.8	15.2
$Adana_{15}$	28	27	1	96.4	3.6
$Adana_{16}$	23	18	0	78.3	0.0
$Adana_{17}$	24	23	21	95.8	87.5
$Adana_{18}$	70	67	23	95.7	32.9
$Adana_{19}$	24	24	12	100.0	50.0
$Adana_{20}$	20	16	0	80.0	0.0
$Adana_{21}$	18	17	11	94.4	61.1
$Adana_{22}$	27	26	6	96.3	22.2
$Adana_{23}$	48	41	6	85.4	12.5
$Ankara_1$	18	18	0	100.0	0.0
$Ankara_2$	44	36	4	81.8	9.1
$Ankara_3$	14	13	2	92.9	14.3
$Ankara_4$	23	23	37	100.0	160.9
$Ankara_5$	61	57	0	93.4	0.0
$Istanbul_1$	8	8	2	100.0	25.0
$Istanbul_2$	13	10	1	76.9	7.7
$Istanbul_3$	14	14	0	100.0	0.0
$Istanbul_4$	11	11	6	100.0	54.5
Total	**911**	**833**	**188**	**91.5**	**20.7**

Table 3.8. Building detection performances on aerial images using decision fusion.

Image Name	**Buildings**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Aerial_1$	17	16	1	94.1	5.9
$Aerial_2$	27	22	34	81.5	125.9
$Aerial_3$	6	6	15	100.0	250.0
$Aerial_4$	9	9	13	100.0	144.4
$Aerial_5$	11	8	9	93.7	21.25
$Aerial_6$	16	15	7	93.7	43.7
$Aerial_7$	9	8	15	88.9	166.6
$Aerial_8$	11	11	8	100.0	72.7
$Aerial_9$	42	32	26	76.2	61.9
$Aerial_{10}$	50	41	25	82.0	53.2
$Aerial_{11}$	47	44	34	93.6	72.3
$Aerial_{12}$	57	51	14	89.5	24.6
$Aerial_{13}$	30	26	27	86.6	90.0
$Aerial_{14}$	19	17	27	89.5	142.1
$Aerial_{15}$	57	46	32	80.7	56.1
$Aerial_{16}$	56	45	8	80.3	14.2
$Aerial_{17}$	44	36	43	81.8	97.7
$Aerial_{18}$	65	56	40	86.1	61.5
$Aerial_{19}$	11	9	8	81.8	72.7
$Aerial_{20}$	16	14	13	87.5	81.2
$Aerial_{21}$	52	48	20	92.3	32.6
Total	**652**	**560**	**419**	**85.8**	**64.2**

have 419 false alarms. These detection results correspond to $TP = 85.8\%$ and $FA = 64.2\%$ respectively. We provide our aerial image set and building detection results of the proposed algorithm in the third column of Figs. 3.24, 3.25, 3.26, and 3.27.

We also provide building detection results on aerial images in Table 3.9. For all methods the false alarm rate is remarkably higher compared to the satellite image detection results. One possible reason for this poor performance is that, in aerial images road segments and nearby land formations resemble buildings due to resolution of these images. Among different local feature vector extraction methods, GMSR has the lowest and Gabor filtering has the highest building detection performance. Besides, decision fusion has a clear advantage compared to all methods (including data fusion) in detecting buildings.

Table 3.9. Building detection performances for aerial test images.

Method	**TP**	**FA**	**TP** (%)	**FA** (%)
Harris	518	366	74.3	52.5
GMSR	479	425	68.7	61.0
Gabor filtering	551	363	79.1	52.1
Data fusion	530	373	76.0	53.5
Decision fusion	**600**	**475**	**86.1**	**68.1**

3.7.3.3. Computation Times

The most important advantage of our local feature based building method is its computation time. Therefore, in this section we consider the time needed by all building detection modules in detail. To note here, as in our previous methods timing directly depends on the test image. As the number of buildings in a test image increases, the number of local feature vectors will also increase. Therefore, our building detection method will need more computation time. To give an idea for the possible reader, we consider the $Adana_1$ test image as a benchmark. We tabulate all CPU timings for each module in Table 3.10. In reporting these results, we used a PC with Intel Core2Due processor with 2.13 GHz. clock speed and has 4 GB of RAM. We used Matlab as our coding platform.

As can be seen in Table 3.10, all modules need similar CPU times. To note here, pre-processing module summarizes image read and variable assignment operations. The total time

Table 3.10. CPU times (in sec.) for building detection operations on the $Adana_1$ test image.

	Local Feature Vector		
Module	Harris	GMSR	Gabor
Preprocessing	0.11	0.16	0.20
Local feature vectors	0.36	0.88	1.67
Kernel density estimation	0.03	1.24	0.63
Building detection	0.53	0.34	0.03
Total	1.02	2.62	2.53

needed to detect buildings (using any local feature vector extraction method) is fairly short. However, depending on the local feature vector extraction step, it changes slightly. If the user needs a faster building detection system, then these slight changes should be taken into account. If we consider data and decision fusion methods, we need 4.80 sec. and 4.67 sec. respectively. Although these methods need more computation times, their building detection performances are also better.

3.7.4. Building Detection Using Steerable Filters

We test our proposed steerable filtering based building detection method on same image set as in the previous section. In the following subsections, we analyze our building detection results on these images quantitatively.

3.7.4.1. Building Detection on Satellite Images

We provide our test images in Figs. 3.18, 3.19, 3.20, and 3.21. In these figures, we provide the detected buildings by steerable filtering for each test image in the fifth column. We tabulate performance of the algorithm on each test image in Table 3.11.

As can be seen in Table 3.11, using steerable filter features we detected 820 of the 911 buildings. We have 124 false alarm. As a result, we obtained TP and FA as 90.0% and 13.6% respectively. Using steerable filtering increased detection performance and decreased false positives dramatically. Therefore, we can claim that using features which can also include structural information increase robustness of the building detection method.

Table 3.11. Building detection performances in Ikonos satellite images using steerable filtering.

Image Name	**Buildings**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Adana_1$	18	17	3	94.4	16.7
$Adana_2$	9	9	1	100.0	11.1
$Adana_3$	31	30	0	96.8	0.0
$Adana_4$	21	21	1	100.0	0.0
$Adana_5$	54	53	3	98.1	5.6
$Adana_6$	47	43	3	91.5	6.4
$Adana_7$	28	27	5	96.4	17.9
$Adana_8$	24	23	1	95.8	4.2
$Adana_9$	24	15	0	62.5	0.0
$Adana_{10}$	14	13	6	92.9	42.9
$Adana_{11}$	21	13	2	61.9	9.5
$Adana_{12}$	32	25	4	78.1	12.5
$Adana_{13}$	67	64	1	95.5	1.5
$Adana_{14}$	33	27	5	81.8	15.2
$Adana_{15}$	28	25	2	89.3	7.1
$Adana_{16}$	23	17	0	73.9	0.0
$Adana_{17}$	24	24	15	100.0	62.5
$Adana_{18}$	70	50	6	71.4	8.6
$Adana_{19}$	24	24	4	100.0	16.7
$Adana_{20}$	20	18	0	90.0	0.0
$Adana_{21}$	18	18	10	100.0	55.6
$Adana_{22}$	27	27	4	100.0	14.8
$Adana_{23}$	48	45	7	93.8	14.6
$Ankara_1$	18	18	4	100.0	22.2
$Ankara_2$	44	37	2	84.1	4.5
$Ankara_3$	14	14	3	100.0	21.4
$Ankara_4$	23	23	30	100.0	130.4
$Ankara_5$	61	60	0	98.4	0.0
$Istanbul_1$	8	8	2	100.0	25.0
$Istanbul_2$	13	7	0	53.8	0.0
$Istanbul_3$	14	14	0	100.0	0.0
$Istanbul_4$	11	11	1	100.0	9.1
Total	**911**	**820**	**124**	**90.0**	**13.6**

3.7.4.2. Building Detection on Aerial Images

We also test our steerable filtering based building detection algorithm on aerial images. Detection results are provided in the last columns of Figs. 3.24, 3.25, 3.26, and 3.27. In Table 3.12, we also tabulate detection results for each aerial image.

Table 3.12. Building detection performance using steerable filtering on aerial images.

Image Name	**Buildings**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Aerial_1$	17	12	0	70.6	0.0
$Aerial_2$	27	21	14	77.8	51.9
$Aerial_3$	6	6	1	100.0	16.7
$Aerial_4$	9	8	4	88.9	44.4
$Aerial_5$	11	10	5	90.9	45.5
$Aerial_6$	16	16	1	100.0	6.3
$Aerial_7$	9	6	3	66.7	33.3
$Aerial_8$	11	11	1	100.0	9.1
$Aerial_9$	42	20	6	47.6	14.3
$Aerial_{10}$	50	23	5	46.0	10.0
$Aerial_{11}$	47	23	12	48.3	25.5
$Aerial_{12}$	57	31	3	54.4	5.3
$Aerial_{13}$	30	28	17	93.3	56.7
$Aerial_{14}$	19	19	2	100.0	10.5
$Aerial_{15}$	57	30	2	52.6	3.5
$Aerial_{16}$	56	28	1	50.0	1.8
$Aerial_{17}$	44	20	8	45.5	18.2
$Aerial_{18}$	65	52	8	80.0	12.3
$Aerial_{19}$	11	10	1	90.9	9.1
$Aerial_{20}$	16	12	3	75.0	18.7
$Aerial_{21}$	52	36	4	69.2	7.7
Total	**652**	**422**	**101**	**64.7**	**15.5**

As can be seen in Table 3.12, we detected 422 of the 652 buildings. We have 101 false alarm. TP and FA ratios are 64.7% and 15.5% respectively. By checking low ratio of false alarms we can conclude that, our steerable filtering based features are robust to small details of

the image. Unfortunately, our detection percentage is decreased comparing with the results of satellite images. In our study region, road edges are very close to building roof edges (they are generally detected as a single edge). Therefore, road edges caused false building edge curvature calculations.

3.7.5. Building Detection Using Color Indices

We tested our color based building detection method on 12 aerial images. These images have $0.3\ m$ resolution. Test images are taken from Istanbul, where buildings generally have red rooftops. Some building detection results can be seen in Fig. 3.28. These examples indicate that our method works fairly well for given aerial test images. We next tabulate building detection performance. Therefore, again we calculate TP and (FA) ratios. In Table 3.13, we tabulate the test results for each image. As can be seen, 162 of the 177 buildings are correctly detected in test images. Unfortunately, some tree clusters in the images are detected as possible buildings by our system. They are possible causes of 25 false positives. As a result, we obtain TP as $\mathbf{91.5}\%$ and FA as $\mathbf{14.1}\%$ for 177 test buildings. Considering this high detection percentage, proposed algorithm gives consistent results for test images.

Table 3.13. Building detection performance on aerial test images using color indices.

Image Name	**Buildings**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Sample_1$	14	14	0	100.0	0.0
$Sample_2$	8	8	0	100.0	0.0
$Sample_3$	3	3	0	100.0	0.0
$Sample_4$	37	34	12	91.9	32.4
$Sample_5$	29	28	0	96.5	0.0
$Sample_6$	7	7	0	100.0	0.0
$Sample_7$	13	9	4	69.2	30.7
$Sample_8$	20	17	6	85.0	30.0
$Sample_9$	10	10	0	100.0	0.0
$Sample_{10}$	24	20	3	83.3	12.5
$Sample_{11}$	6	6	0	100.0	0.0
$Sample_{12}$	6	6	0	100.0	0.0
Total	**177**	**162**	**25**	**91.5**	**14.1**

Figure 3.28. Sample building detection results using color indices in aerial images. First column: $Sample$ test images (1,2,4,10,11,12). Second column: building detection results

3.7.6. Damaged Building Detection Using Shadow Information

In this part, we give experimental results of our damaged building detection method. Unfortunately, we do not have many damaged buildings in our test image database. Therefore, we analyze experimental results on two damaged buildings.

In this section, we provide two test images, one containing undamaged buildings ($Damage_1$) and the other containing damaged buildings ($Damage_2$). Detected buildings in $Damage_1$ test image are given in Fig. 3.29. Damage measures, r, of these buildings are calculated as follows. On the upper side only first three buildings are detected, and their damage measures are calculated as 1.94, 1.79, and 2.26 respectively. For the buildings laying horizontally in the center of the image, damage measures are calculated as 1.79, 1.79, 2.34, 2.38, 2.70, 2.35, 2.31 respectively. Finally, for the building on the lower left side of the image, the damage measure is calculated as 2.23. It can be seen that the obtained damage measures of these buildings are very similar. The average of these damage measures is calculated as 2.17. Since the user knows that all of these buildings are healthy, the degree of the damage on other buildings can be estimated by comparing their damage measures with the average value 2.17. After calculating the damage measures

Figure 3.29. Detected buildings in $Damage_1$ test image. (Undamaged building measures are calculated on these detected buildings)

for the undamaged building set (selected by the user), we use the $Damage_2$ image that contains damaged buildings in order to test our algorithm as given in Fig. 3.30. For the building which is on the upper left side of this test image, damage measure is 1.6. The second building on the upper side could not be found by our method, so damage degree could not be measured. For the building on the upper left side of the image, the damage measure is obtained as 1.6 again. By

comparing with the average value 2.17, we can say that these two buildings are undamaged. For the building on the lower left side of the image, the damage measure is calculated as 2.78. This result is also similar to damage measures of undamaged buildings, and that indicates this building is also undamaged. On the lower side of the image, damage measures of last three buildings are calculated as 7.75, 4.22, and 4.67 respectively. These values are very high compared to the average value 2.17. Therefore, these values can give an idea to user about the damage in these buildings. However we could make very few experiments on damaged buildings, our experi-

Figure 3.30. Detected buildings in $Damage_2$ test image. Detected damaged buildings are labeled with a star

mental results show importance of developed method. First, most of the studies in the literature needs prior information about region, however developed method does not need previous images of the disaster region. Second, proposed method needs very low computation time and may help to human observers for fast damage disaster. Deficiency of proposed method is that; the system may not be used if shadow regions are not detected robustly.

3.7.7. Comparison of the Proposed Building Detection Methods

After presenting experimental results of each building detection method in detail, we finally compare all building detection methods presented in this chapter in terms of detection performance, computation times, and robustness.

In SIFT and graph theory based building detection method (Section 3.2), we obtained 88.4% detection performance and 16.5% false alarms. This performance on our satellite images with different building characteristics is very encouraging. Unfortunately, this method may not detect buildings if the contrast between their rooftop and the background is low. Besides, closely spaced buildings may be detected as one single building. It always needs template buildings to find similarities in given test image. In addition, finding these similarities by sub-graph matching requires high computation time. Lastly, SIFT based detection algorithm can not be used on lossy compressed images. Thus, we could not test algorithm on our aerial image data set. These are the main drawbacks of the system.

In order to overcome drawbacks of the SIFT and graph theory based algorithm, we developed local feature based probabilistic building detection method. We first provide the specific Gabor filtering based local features. In the same Ikonos image data set, we obtained $TP = 86.1\%$ with $FA = 19.4\%$. Since Gabor features are robust to lossy compression effects, we also tested algorithm on our aerial test image data set. There, we obtained $TP = 88.2\%$ with $FA = 66.1\%$ false alarms. However in Ikonos image data set detection performance decreased and false alarms are increased slightly, Gabor feature based method has impressive timing requirements. To compare with SIFT based algorithm we can choose $Adana_8$ test image. On this image, SIFT based algorithm detected buildings in 269.44 sec. Whereas, for the same test image Gabor based method detects buildings in 26.05 sec. This very low timing requirement makes Gabor based method suitable for real time applications such as unmanned aerial vehicles. Furthermore, Gabor filter based method does not require building samples from study region in order to detect other buildings. Fixing Gabor filter parameters is enough to adapt the algorithm to a different kind of remotely sensed image. The only disadvantage of Gabor based method is that it may not detect a building if it is very dark compared to background.

In feature fusion based method, we obtained best detection performance in decision fusion method. After extensive testings, we observed that our fusion based method is able to detect most of the buildings (having different size, shape and intensity values) in a correct manner on both aerial and satellite images. In our panchromatic Ikonos image data set we obtained $TP = 91.5\%$ with $FA = 20.7\%$. In grayscale aerial image data set, we obtained $TP = 85.8\%$ with $FA = 64.2\%$. Although, in fusion based method FA in aerial images decreased slightly, it is still very high. Therefore, to cope with very high resolution and noise effects in aerial images more robust algorithms are needed.

In all of these building detection methods we observed that more robust features are needed to decrease false alarms (especially for aerial images). To increase robustness of building detection method to false alarms, we developed steerable filtering based features. Since these novel features also contain structural information, we decreased number of false alarms which occur because of noise or small changes on ground surface. In our panchromatic Ikonos image dataset, we obtained TP and FA as 90.0% and 13.6% respectively. Since these structural features are also robust to compression effects, we could also test algorithm on grayscale aerial images. In our aerial image data set we obtained TP and FA as 64.7% and 15.5% respectively. Comparing with feature fusion based algorithm we decreased false alarms in aerial images impressively. Unfortunately, this also decreased the detection performance.

To overcome the difficulty of building detection in aerial images, we used color indices. On a different color aerial image dataset we obtained $TP = 91.5\%$ with $FA = 14.1\%$. These results indicate that; if color information is available, it can provide very useful information to increase detection performance. Besides, by developing the color indices based approach, we also detected damaged structures.

3.8. SUMMARY OF THE CHAPTER

In this chapter, we introduced novel methods for building detection in very high resolution satellite and aerial images. First, we started with developing our SIFT based urban region detection approach that we introduced in Section 2.2. There, we used SIFT algorithm to extract features of both template and test images, and we presented these features as a graph. Then, we developed multiple subgraph matching to detect urban region subgraph in test image. Based on detected subgraph we extracted urban region boundaries. In Section 3.2, we introduced a novel graph cut method to cut urban region subgraph. After cutting the urban region subgraph, we detected separate buildings. For satellite images the building detection performance is very high.

To overcome drawbacks of SIFT based building detection algorithm, we introduce a novel method in Section 3.3. We used a probabilistic framework to detect building centers. To do so, we defined the spatial coordinates of buildings (to be detected) as joint random variables. We formed their pdf by the nonparametric variable kernel density estimation method. In estimating the pdf, we used local feature vectors (extracted from the image) as observations. Then, we detected building locations using the modes of the estimated pdf and other probabilistic constraints. After

extensive testings, we observed that our method is able to detect most of the buildings (having different size, shape and intensity values) in a correct manner on both aerial and satellite images. However, for aerial images the false alarm rate is fairly high.

In order to decrease false alarm rates, we proposed a more robust feature in Section 3.4. There, we used steerable filters to extract structural features. We detected buildings regarding curvature of these features. Using this structural information increased robustness of our algorithm to noise and redundant details in image. Using steerable filtering based features, we obtained the lowest false alarm rate in aerial image data set.

Detection performance in aerial images was always problematic in the proposed methods. In order to increase the detection capability, we also proposed a novel building detection method using color invariant features in Section 3.5. We used color invariant features to extract red-rooftops and shadows. We verify the appearance of buildings (having red and non-red rooftops) using shadow information. As a final step, we developed a novel method (box-fitting) to determine building shapes.

In Section 3.6, we developed the same system to detect damaged buildings. We defined a measure to estimate damage degree using rooftop and shadow segments for each building. Most important feature of the algorithm is that it works without pre-knowledge about study region. Results indicate the possible use of system in disaster management and military applications.

In Section 3.7, we presented experimental results of each building detection method on real Ikonos satellite and aerial images. We also tabulated detection performances for each test image in tables. After introducing each building detection method in detail, in Section 3.7.7 we compared building detection methods in terms of performance, timing requirements and computation times.

As a result, we can conclude the chapter with presenting following observations. First, one can use SIFT based building detection method (Section 3.2), if high performance is desired and if the contrast difference of the background and buildings are very high. If slightly lower performance is tolerable, fusion of local feature vectors can be preferred (Section 3.3). The steerable filtering based method (Section 3.4) will be best choice if study region is very complex. For color aerial images, color index based method can also be used for damage detection purposes.

Unfortunately, this method may not work if the color of terrain is close to red.

4. ROAD DETECTION

Urban regions are dynamic environments and road maps change frequently. Therefore, automatic detection of roads from very high resolution aerial and satellite images is a very important research field. Although road segments have simpler shapes compared to buildings, detecting them in remotely sensed images is more difficult. The main problem in detection is the occlusion. Roads are generally occluded by other nearby objects like buildings and trees. Road segments may also have different colors and their widths may change. In addition to that, junctions of unknown number of roads, roundabouts, and other road characteristics may increase the difficulty of the problem. Therefore, advanced methods are needed to detect road segments in aerial and satellite images. In this chapter, we propose novel methods for automatic road segment detection in very high resolution aerial and satellite images. Before presenting our methods, we start investigating previous studies on road detection.

4.1. PREVIOUS STUDIES ON ROAD DETECTION

Some researchers developed semi-automatic techniques to detect roads [82, 83, 84]. Yang and Wang [85] developed a method to detect main roads from satellite images. First, they detected road primitives such as straight lines and homogenous regions. Then, they linked detected primitives. Unfortunately, their method can not detect urban roads and occluded road segments. Ma *et al.* [86] detected parallel edges in panchromatic ETM (Enhanced Thematic Mapper) images to detect road segments. They linked discontinuous road segments using perceptual organization rules. Dell'Acqua and Gamba [17] detected road networks and built areas in SAR images. They extracted straight edges and detected built areas with a clustering based approach. They assumed longitudinal gaps as road networks. Rianto *et al.* [87] proposed an approach to detect main roads in SPOT satellite images. For this purpose, they detected Canny edges and classified straight line segments using the Hough transform. They assumed straight and parallel line segments as roads. Unfortunately, this approach can not be sufficient alone to detect curvilinear and complex roads in urban scenes. Mayer *et al.* [88] and Ünsalan and Boyer [5] provide excellent surveys on road detection in aerial and satellite images.

Some researchers developed algorithms to detect both buildings and roads. Ünsalan and Boyer [5] detected separate buildings and street networks from multispectral satellite images.

Their method depends on using vegetation indexes, clustering, decomposing binary images, and graph theory. Sarkar and Boyer [9] used graph theory to detect edges which obey perceptual organization rules to detect building and road edge segments in panchromatic satellite images. Akçay and Aksoy [58] proposed a novel system for detecting built areas and roads using unsupervised segmentation in high resolution satellite images. Montesinos and Alquier [89], developed a novel approach to detect thin objects in noisy images. They performed perceptual grouping on edge segments using active contours. They tested validity of their approach by detecting roads in aerial images and detecting blood vessels in medical images.

4.2. ROAD DETECTION USING STEERABLE FILTERS

In this section, we propose a new approach to detect main roads automatically from very high resolution panchromatic satellite images in a fast and robust manner. For this purpose, first we extract local structural features using steerable filters as in our building detection approach (Section 3.4). Then, we group these features using basic laws of perceptual organization [90]. We detect missing road segments by a simple tracking approach. Finally, we present experimental results of proposed approach on Ikonos satellite images in Section 4.4.

4.2.1. Preprocessing

Due to very high resolution of satellite images, edge detectors may not work properly to detect road edges. They can also detect very small noisy terms in the image. These redundant edges may give rise to false detection and increase the complexity of the problem. To prevent this drawback, we filter the grayscale satellite image with the bilateral filter introduced in Section 2.2.1. After filtering, we obtain $I_b(x, y)$.

4.2.2. Extracting Road Features using Steerable Filtering

In order to extract discriminative local road features, we follow a similar approach to steerable filtering based building detection algorithm in Section 3.4. By filtering our $I_b(x, y)$ test image with a steerable filter in θ direction, we obtain $J_\theta(x, y)$. In $J_\theta(x, y)$, we expect edges laying perpendicular to the filtering direction to have higher values. Therefore, by thresholding $J_\theta(x, y)$ we obtain representative object edges which are perpendicular to the filtering direction. In our extensive tests, we observed that choosing the threshold value as 20% of maximum magni-

tude in $J_\theta(x, y)$ is suitable to extract road edges. As a result, we obtain a binary image, $F_\theta(x, y)$ that holds locations of road features.

In this study, we pick our steerable filtering directions as $\theta \in [0, \pi/12, ..., 23\pi/12]$. Therefore, we have a total of 12 filters. Since we do not have a prior knowledge about the road alignments, these different directions are necessary. We extract local feature vectors in all θ directions. In $F_\theta(x, y)$ we eliminate some complex shaped features since they can not represent road edges. Therefore, we check the compactness of each feature separately. Compactness is defined as the ratio of a region's area to the square of its perimeter, normalized by 4π. We eliminate features having $compactness < 4\pi/60$ as in [5].

We pick the $Road_1$ test image to explain our algorithm. In Fig. 4.1 (a), we present the original $Road_1$ test image. In Fig 4.1 (b), we represent all detected road features ($F_\theta(x, y)$ where $\theta \in [0, \pi/12, ..., 23\pi/12]$). In Fig 4.1 (c), we represent the road features after the compactness operation. We present features over the test image to increase visibility.

(a) Test image (b) Detected road features (c) After the compactness test

Figure 4.1. Original $Road_1$ test image, detected road features before and after the compactness test

4.2.3. Grouping Extracted Features to Detect Road Centers

Road edges are generally symmetric and close to each other. They generally have good continuation. Considering these characteristics, road features may be grouped. Herein we benefit from perceptual organization laws to group detected features in $F_\theta(x, y)$. Then, using grouped features we detect road segment centers.

To group road features, we use $F_{\theta_1}(x, y)$ and $F_{\theta_2}(x, y)$ matrices where $\theta_2 = \theta_1 + \pi$. There-

fore, when $F_{\theta_1}(x, y)$ holds features of one side of the road segment, $F_{\theta_2}(x, y)$ holds features of the other side of the same road segment.

To group mutual road features in $F_{\theta_1}(x, y)$ and $F_{\theta_2}(x, y)$, we benefit from morphological operations in order to reduce computation time and complexity. Assume that we have a total of K features in $F_{\theta_1}(x, y)$ denoted as $F^k_{\theta_1}(x, y)$ for $k \in [1, 2, ..., K]$. Also assume that we have a total of L features in $F_{\theta_2}(x, y)$ denoted as $F^l_{\theta_2}(x, y)$ for $l \in [1, 2, ..., L]$. To obtain the associated features in $F_{\theta_1}(x, y)$ and $F_{\theta_2}(x, y)$ we pick each feature from these matrices separately. To explain the following steps clearly, we pick an example as in Fig. 4.2 by zooming in to our $Road_1$ test image. To check whether these two features are associated, we first apply the dilation operation with a

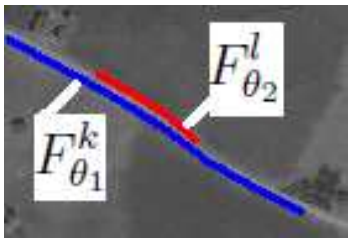


Figure 4.2. Zoomed $Road_1$ test image and sample $F^k_{\theta_1}$ and $F^l_{\theta_2}$ features on it

disk of radius 10 pixels [91]. We then obtain $D^k_{\theta_1}(x, y)$ and $D^l_{\theta_2}(x, y)$. We then check their intersection as

$$C_{k,l}(x, y) = D^k_{\theta_1}(x, y) \cap D^l_{\theta_2}(x, y) \tag{4.1}$$

if $C_{k,l}(x, y) \neq \emptyset$, we label these two features as associated. We assume a portion of the road segment to be laying between $E^k_{\theta_1}(x, y) = C_{k,l}(x, y) \cap F^k_{\theta_1}(x, y)$ and $E^l_{\theta_2}(x, y) = C_{k,l}(x, y) \cap F^l_{\theta_2}(x, y)$. We label the center of this road segment by $C_{k,l}(x, y)$. We can approximately calculate the width of this road segment as

$$w_{k,l} = \frac{1}{N} \sum_{i=1}^{N} dist(E^k_{\theta_1}(x_i, y_i), E^l_{\theta_2}(x, y)) \tag{4.2}$$

where, N is the total number of pixels in $E^k_{\theta_1}$. Here $dist(\cdot, \cdot)$ is the minimum Euclidean distance between the point $E^k_{\theta_1}(x_i, y_i)$ and $E^l_{\theta_2}(x, y)$. In some cases, one feature can be longer than

$C_{k,l}(x,y)$. As can be seen in Fig. 4.2, some part of the $F^k_{\theta_1}(x,y)$ feature will not intersect with $F^l_{\theta_2}(x,y)$. Therefore, this part will not contribute to form the road segment. We would like to use this non-intersecting part in road segment extraction. Therefore, we benefit from $C_{k,l}(x,y)$ and $w_{k,l}$. Let's call the non-intersecting part of $F^k_{\theta_1}(x,y)$ as $G^k_{\theta_1}(x,y) = F^k_{\theta_1}(x,y) \setminus E^k_{\theta_1}(x,y)$. To estimate the corresponding road center, we calculate two normal vectors for each pixel in $G^k_{\theta_1}(x,y)$. We multiply these two normal vectors by $w_{k,l}/2$. We assume end points of two normal vectors as $(x_n(i), y_n(i))$ where $i \in [1,2]$. We pick the end point $(x_n(i), y_n(i))$ which has smaller Euclidean distance to $C_{k,l}(x,y)$, and set $C_{k,l}(x_n(i), y_n(i)) = 1$.

After applying the same method to each $F^k_{\theta_1}(x,y)$ and $F^l_{\theta_2}(x,y)$ for $k \in [1,2,...,K]$ and $l \in [1,2,...,L]$ respectively, we detect the road segments for θ_1 as

$$C_{\theta_1}(x,y) = \bigcup_{k,l} C_{k,l}(x,y) \tag{4.3}$$

We finally obtain all the road segments in 12 different θ directions using the above method. Combining them, we obtain the final road segment from the test image as $C(x,y)$. In Fig. 4.3, we provide the detected road segments for our $Road_1$ sample image. As can be seen in this figure, most of the road segments are detected correctly. Unfortunately, we could not detect junctions and very small road segments. To recover these missing road segments, next we present a road tracking method.

Figure 4.3. Detected road centers from the $Road_1$ test image by feature grouping method

4.2.4. Road Tracking

Sometimes road segments may be occluded by other objects like buildings and trees. The contrast between the road and the nearby region may be very low. In these cases, some of the road

segments may not be detected by our previous approach. Also, road junctions and roundabouts may not be detected because of their complex shapes. To detect these missing road segments, we propose using a prediction based road tracking algorithm in this section.

For tracking, first we obtain skeleton of detected road segments matrix $C(x, y)$ [39]. Then, we detect the endpoints of this skeleton. Endpoints are chosen as two pixels which have only one neighbor pixel and which have highest Euclidean distance between each other. Assume that, we obtained endpoints as $(x_p(h), y_p(h))$ where $h \in [1, 2, ..., H]$. Generally road segments do not end suddenly. Therefore, we assume that there is a missing road segment starting from each $(x_p(h), y_p(h))$ pixel. It can be used to start tracking in order to extract missing road segment.

Before starting the tracking process, we should estimate the tracking direction. For this purpose, we use 10 road pixels in $C(x, y)$ which form a connected segment with $(x_p(h), y_p(h))$ pixel. We fit an ellipse to these points. We calculate the tracking orientation as the angle (θ_t) between horizontal axis and the major axis of the fitted ellipse. Our tracking algorithm works iteratively until a possible road segment can not be found in the tracking direction. At the first step of iteration, we assume tracking direction as equal to θ_t. At each iteration, we calculate θ_t angle again.

After determining the tracking direction, we estimate the next road pixel in tracking direction. Since roads generally do not have sharp turns, we only consider three neighbor pixels to $(x_p(h), y_p(h))$ in θ_t direction as the next road pixel candidates. We calculate x and y coordinates of candidate pixels as; $x_c(n) = x_p(h) + (cos(\theta_t) + (n-2)\frac{\pi}{4})$ and $y_c(n) = y_p(h) + (sin(\theta_t) + (n-2)\frac{\pi}{4})$ respectively. Here $(x_c(n), y_c(n))$ $n \in [1, 2, 3]$ are candidate road pixels in tracking direction. We illustrate candidate road pixels in Fig. 4.4. In Fig. 4.4, each box

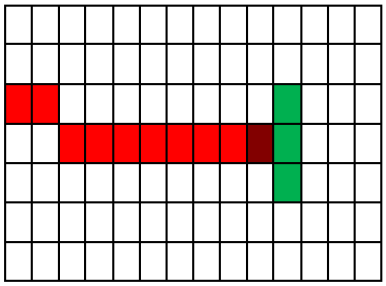


Figure 4.4. Tracking example. Red pixels indicate 10 pixels belong to previously detected road segment, and green pixels indicate new road pixel candidates

represents a pixel in the image. Red pixels show 10 road pixels in $C(x, y)$ that are connected to $(x_p(h), y_p(h))$. $(x_p(h), y_p(h))$ which is represented by a darker red. Three green pixels are $(x_c(n), y_c(n))$ $n \in [1, 2, 3]$ candidate pixels.

Now, the strongest candidate should be chosen to continue tracking. Since we know approximate tracking direction (θ_t), in another saying since we know approximate direction of road segment, we can use $J_\theta(x, y)$ filter responses to find the strongest candidate. To speed-up the algorithm, we do not check each $J_\theta(x, y)$ for 12 different directions to find the strongest candidate. We know that, the road segment is laying in θ_t direction. Therefore, each pixel of road segment will give similar response when we filter image in $(\theta_t + \pi/2)$ direction. Therefore, we only use $J_{(\theta_t+\pi/2)}(x, y)$ filter response to choose the strongest road candidate. First, we assume that ten road pixels in $C(x, y)$ matrix that are connected to $(x_p(h), y_p(h))$ are presented with $(\widehat{x}_p(m), \widehat{y}_p(m))$ (where $m \in [1, 2, ..., 10]$). We calculate mean of their responses in $(\theta_t + \pi/2)$ filtering direction as

$$\mu = \frac{1}{10} \sum_{m=1}^{10} J_{(\theta_t+\pi/2)}(\widehat{x}_p(m), \widehat{y}_p(m)) \tag{4.4}$$

Calculating $S(n) = |J_{(\theta_t+\pi/2)}(x_c(n), y_c(n)) - \mu|$ for $n \in [1, 2, 3]$, we obtain the similarity of filter responses of candidate pixels to the filter responses of previous road pixels. We select $(x_c(n), y_c(n))$ pixel which gives the smallest $S(n)$ value as our next road pixel in tracking. Then, we assume the selected $(x_c(n), y_c(n))$ pixel as our next end point and estimate the new candidate pixel. This tracking method is applied iteratively to extract the missing road segment. When missing road segment is extracted completely, iteration should be stopped. Therefore, we use a control threshold $c_{th} = 10$. At the each step of the iteration, we control if similarity measure satisfy $\mu < c_{th}$, otherwise we stop iteration and pick another $(x_p(h), y_p(h))$ endpoint to start tracking.

In Fig. 4.5, we present detected roads in our $Road_1$ test image by tracking approach with green labels. In Section 4.4, we analyze our road detection method on panchromatic Ikonos satellite images quantitatively. Next, we present another road detection approach based on color information.

Figure 4.5. Detected road centers using tracking (labeled with green)

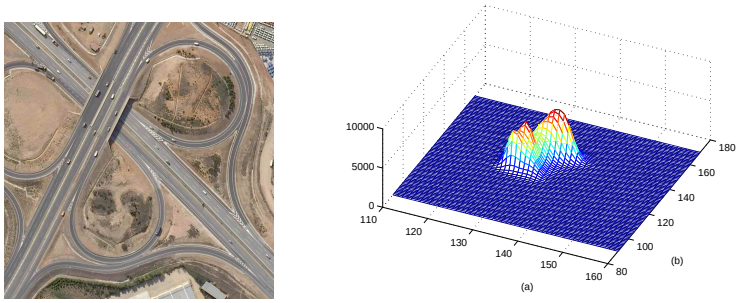
4.3. ROAD DETECTION USING COLOR INFORMATION

Besides using structural features as in the previous section, color information (if available) may also give important cues to detect asphalt road segments. Therefore, in this section we introduce a new algorithm to detect asphalt road segments using their color information. To do so, we introduce a color feature and benefit from one class classification method.

4.3.1. Training the Classifier with Invariant Color Features

In many computer vision applications, the CIELab color space is used since it mimics the human visual system [92]. CIELab color space bands enhance different colors best and minimize color variances. After transforming the RGB image into CIELab color space, we again obtain three bands as L, a, and b [93]. Here, L band corresponds to intensity of the image pixels. a, b bands contain chroma features of the image. They give information about the color independent of illumination. In the literature, many researchers used Euclidean distances of L, a, and b bands of images to find similar regions generally for segmentation purposes [94]. We apply a similar methodology to segment out the asphalt road segments from the image in this section.

In order to detect asphalt roads in given test image, first we constructed a training set. We cut 10 asphalt road patches from several color aerial images manually. We applied the CIELab transform to all of these patches, and stored the a and b band values for each pixel. We did not use the L band values, since we want our training set to be invariant from illumination. We store the chroma features in our data base as $f_j = [a_t(j), b_t(j)]$ where $j \in [1, 2, ..., J]$. Here, J is total number of pixels in 10 asphalt patches. For a sample test image $Asphalt_1$, we provide the distributions of both road pixel and non-road pixel values in Fig. 4.6.



(a) Original image (b) Pixel distributions

Figure 4.6. The $Asphalt_1$ test image and the pixel distributions in the a, b color space

4.3.2. Detecting Asphalt Roads Using One Class Classifier

In most classification problems, the label for each class is known. In remotely sensed images, we have unknown number of classes (such as trees, buildings, parks, roads, agricultural fields, etc.). If we have only samples of one class as in our problem, classification can not be done by a classical two-class classifier, since we can not represent all other classes in one class. This is also the case for our asphalt road detection problem. Fortunately, a one class classifier can be used to separate this small class from other classes [95].

The problem in one class classification is to make a description of a training set features and to detect which new feature resemble this training set. The difference with conventional classification is that; in one class classification only examples of one class are available. The features from this class will be our training set. All of the other irrelevant features will be outliers. Therefore, real problem is defining a boundary around the target class. In the related literature, Erçil and Büke [96] fitted a surface around the feature point cloud in feature space by implicit surface fitting. They labeled points inside this surface.

To achieve a faster system, we represent the training set with its mean and covariance value. Then we classify the test samples using the Mahalanobis distance [97]. Different from previous approaches, for a given test image we calculate the Mahalanobis distance for all pixels. To classify them, we calculate Otsu's threshold as a class boundary. For the $Asphalt_1$ test image, we provide the detected road pixels using one class classifier in Fig. 4.7.

Figure 4.7. Detected asphalt road pixels from the $Asphalt_1$ test image

4.4. EXPERIMENTAL RESULTS

In this section, we present experimental results of our two road detection methods. After representing results on real test images and tabulating their performances, we discuss important properties and deficiencies of both methods. Next, we start with the experimental results of steerable filtering based road detection method.

4.4.1. Road Detection Using Steerable Filters

We test steerable filtering based road detection approach on 8 Ikonos satellite images. We provide road detection results on our data set in Figs. 4.8 and 4.9. In these figures, we provide the original images in the first column, and detected roads in the second column. Here, red labels indicate extracted road segments using steerable filtering based features and green labels indicate road segments extracted by the tracking approach. We tabulate performance of the algorithm on each test image in Table 4.1. In this table, we provide the total length of roads (in pixels) for each test image under the 'RL' column.

As can be seen in Table 4.1, on 8 test images our road detection approach detected 14263 of the 19752 road pixels correctly. This corresponds to a 72.21% detection rate. There are also 8320 false alarms and this corresponds to a $FA = 42.12\%$ rate. TP percentage of our road detection method is satisfactory for the given Ikonos satellite images. Among these test images, the best performances are obtained for $Road_2$, $Road_4$, and $Road_5$ images. In these images, intensity differences between road segments and background is very high. Lowest performance is obtained for the $Road_8$ test image. It represents the most complex scene with many buildings and road junctions. The highest false alarm rate is obtained for the $Road_4$ image. In this image, edges of agricultural fields lead to false alarms.

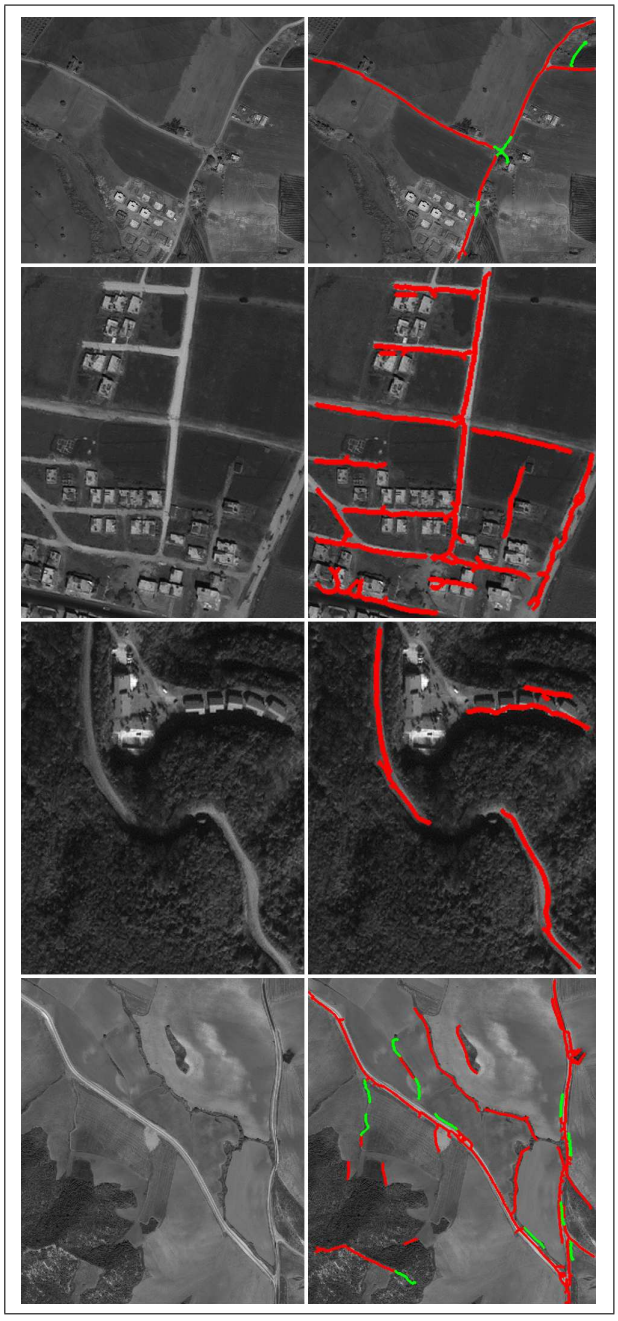


Figure 4.8. Road detection results with steerable filtering based algorithm for *Road* images (1 to 4) for each row separately. First column: original test images; second column: detected road pixels (red labels show detected road segments with steerable filter features, and green labels show tracking results)

Figure 4.9. Road detection results with steerable filtering based algorithm for *Road* images (5 to 8) for each row separately. First column: original test images; second column: detected road pixels (red labels show detected road segments with steerable filter features, and green labels show tracking results)

Table 4.1. Road detection performances of the steerable filtering based method on Ikonos satellite images.

Image Name	**RL**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Road_1$	897	886	13	98.77	1.45
$Road_2$	1519	1519	538	100.00	35.42
$Road_3$	573	502	234	87.61	40.84
$Road_4$	1758	1758	2462	100.00	140.05
$Road_5$	1447	1529	1636	100.00	113.06
$Road_6$	2160	1754	804	81.20	37.22
$Road_7$	725	648	173	89.38	23.86
$Road_8$	10673	5667	2460	53.10	23.05
Total	**19752**	**14263**	**8320**	**72.21**	**42.12**

We tabulate calculation times needed for road detection steps for the $Road_1$ test image in Table 4.2. As can be seen in this table, road network in the $Road_1$ image is extracted in 98.82 sec. This computation time is very encouraging for road detection in very high resolution satellite images.

Table 4.2. CPU times (in sec.) for steerable filtering based road detection on the $Road_1$ test image.

Operation	**Time**
Feature extraction	26.92
Feature grouping	2.85
Road detection	38.65
Tracking	30.40
Total	**98.82**

4.4.2. Road Detection Using Color Information

In this section, we test our color based road detection approach on color satellite and aerial images. First, we provide road detection results on 4 color satellite images that we used in Section 4.4.1. We provide road detection results on our data set in Fig. 4.10. In this figure, we

provide the original color satellite images in the first column, and detected roads for each test image in the second column. Here, red labels indicate road pixels detected using color features. We also present detection performances for each test image in Table 4.3. As can be seen in this table, our method has $TP = 66.88\%$ with $FA = 25.28\%$. Unfortunately, the performance is not satisfactory here.

Table 4.3. Road detection performances of color feature based method on satellite images.

Image Name	**RL**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Road_1$	5336	4525	2873	84.80	53.84
$Road_2$	8243	7780	2584	94.38	31.34
$Road_3$	5246	3406	227	64.92	4.32
$Road_4$	34589	20014	7823	57.86	22.61
Total	**53414**	**35725**	**13507**	**66.88**	**25.28**

The best detection performances are obtained in $Road_1$ and $Road_2$ test images. In these two images color of road pixels are distinguishable from nearby regions. In addition to that, we have used some of the road pixels of these test images in training set. Therefore, we obtained higher performances on these images. In $Road_3$ test image, one segment of the road was missing at the middle of the road with steerable filtering based filter (in Section 4.4.1). However, using color features we detected it completely. The reason of this result is that; in the panchromatic $Road_3$ test image we could not detect any strong edge at the middle of the road, so we could not detect this part. However, we can detect the asphalt color in this region. Therefore, we also detected this part of the road segment successfully in our color feature based method. Unfortunately, the performance on $Road_4$ test image is not very promising. The reason of high false alarms and low detection performance is the similar color of nearby pixels to the road segment. The calculation time of the proposed method is very encouraging. For the $Road_1$ test image, road detection using color features takes only 8.33 sec.

We finally test our color based road detection approach on 5 color aerial images. We provide the road detection results on our data set in Fig. 4.11. In this figure, we provide original images in the first column, and detected roads for each test image in the second column. Here, red labels indicate road pixels which are extracted roads using color features. We present detection performances for each test image in Table 4.4. As can be seen in this table, the proposed method has $TP = 85.10\%$ with $FA = 23.17\%$. Although we do not use any structural feature in this

Figure 4.10. Road detection results with color feature based algorithm for color *Road* images (1 to 4) for each row separately. First column: original color satellite test images; second column: detected road pixels

Figure 4.11. Road detection results with color feature based algorithm for $Asphalt$ images (1 to 5) for each row separately, first column: original aerial test images; second column: detected road pixels

method, detection performance is fairly high for these test images.

Table 4.4. Road detection performances of color feature based method on aerial images.

Image Name	**RL**	**TP**	**FA**	**TP** (%)	**FA** (%)
$Asphalt_1$	316908	272119	31337	85.87	9.89
$Asphalt_2$	88210	85539	42159	96.97	47.79
$Asphalt_3$	52879	37444	3256	70.81	6.16
$Asphalt_4$	154842	136905	56220	88.42	36.31
$Asphalt_5$	52353	34052	21146	65.04	40.39
Total	**665192**	**566059**	**154118**	**85.10**	**23.17**

4.4.3. Comparison of the Proposed Road Detection Methods

In Section 4.2 we developed a new method to detect road segments in very high resolution panchromatic satellite images automatically. There, we used basic rules of perceptual organization. For this purpose, first we extracted local structural features using steerable filters as in our building detection approach (Section 3.4). Then, we grouped these features using basic laws of perceptual organization. We detected missing road segments by a new road tracking approach. In 8 test images, our road detection approach detected 14263 of 19752 road pixels correctly, which corresponds to a 72.21% detection rate. There are also 8320 false alarms and this corresponds to a $FA = 42.12\%$. TP of the proposed road detection method is satisfactory for the given very high resolution panchromatic Ikonos images. The best detection performances are obtained when intensity differences between road segments and background are very high. Unfortunately, complex buildings and edges of agricultural fields led to false detections. These false alarms may be eliminated if multispectral or color information is used.

In Section 4.3, we introduced a new method to detect asphalt roads using color information. For this purpose we used training features belonging to asphalt road patches. We then detected pixels having similar values in the given test image. We tested our method on 5 color aerial images. We obtained $TP = 85.10\%$. TP of the proposed method is satisfactory for asphalt road detection. We obtained the FA as 23.17%. We also tested our color feature based method on 4 color satellite images that we used in Section 4.2. In these images, we obtained $TP = 66.88\%$ and $FA = 25.28\%$. Since we do not detect the same entities in both methods, it is not possible to compare them by their performances. However, there are some observations based on the

performed tests. First, in steerable feature based method we can miss some of road segments if their edges are not very strong. However, color feature based method can detect these missing segments easily. Unfortunately, color feature based method detects too many false alarms if nearby regions has very similar color to the road pixels. In order to increase performance, color features can be fused with steerable filtering features.

4.5. SUMMARY OF THE CHAPTER

In this chapter, we introduced two new methods to detect roads automatically on panchromatic satellite and color aerial images. In Section 4.2 we proposed a new approach to detect roads in panchromatic Ikonos satellite images. In order to detect roads in a robust manner, we used structural features. We extracted these features using a set of steerable filters. We used the idea of perceptual organization to group detected features. After detecting feature couples (two mutual edge structures of a road segment), we detected road centers. In order to recover missing road segments, we introduced a road tracking approach. Although our detection performance is satisfactory, straight and long edges of a terrain (like edges of agricultural fields) can cause false alarms.

Unfortunately, our steerable filtering based method does not provide good results on complex aerial scenes. Thus, in Section 4.3 we introduced another method to detect asphalt roads using their color information. For this purpose, we introduced a one class classifier based method. We first constructed a training set which includes color features of manually extracted asphalt road regions. We used CIELab color space to extract color features. We also extracted color features of given test image. Using training features and a one class classifier, we detected asphalt roads in aerial and satellite images. Although the classification results are encouraging, the proposed method may be improved further by fusing color and structural features.

5. CONCLUSIONS

In this dissertation, we proposed novel approaches to detect man-made objects in very high resolution satellite and aerial images. Specifically, we aimed to detect urban regions, separate buildings and road segments. For this purpose, we developed novel approaches using local and semi-local invariant features. These features are very powerful in detecting objects under various imaging conditions (like illumination, viewing angle, etc.). Unfortunately, extraction of invariant features is not sufficient for detecting objects. Therefore, we formalized the problem by developing graph theoretical, probabilistic, region and perceptual organization based methods. The aim here is to extract structural information to verify object appearance. We presented our approaches for each urban object detection problem (urban region detection, building detection, and road detection) in a separate chapter. At the beginning of each chapter, we reviewed the related literature. At the end of each chapter, we evaluated and compared our approaches in terms of performance, timing, and complexity.

In Chapter 1, we briefly introduced satellite and aerial images. We also provided the background of the urban object detection problem. We introduced local and semi-local invariant features for robust object detection. We observed that, many local features are used to detect different kind of objects in the literature, but they are not used to detect man-made objects in remotely sensed images. Thus, we decided to use local and semi-local features for robust object detection in very high resolution satellite and aerial images.

In Chapter 2, we introduced two novel methods to detect urban region boundaries on gray scale very high resolution satellite and aerial images. First, we introduced novel SIFT feature based method in Section 2.2. There, we picked two template building images, and obtained their SIFT features. We also obtained the SIFT keypoints for the test image. Then, we introduced novel multiple subgraph matching approach to match template and test images SIFT features. Detected subgraph in the test image provided us the urban region in test image. One important feature of this method is that, one can generalize it to detect any kind of object in satellite images. Our SIFT based algorithm has high performance using only grayscale information. It has also some deficiencies. In order to use the proposed algorithm, a user should prepare a template building data set. Calculation time increases exponentially when complexity and size of image increases. As another deficiency, we observed that SIFT based algorithm does not work well on

lossy compressed images.

To overcome drawbacks of SIFT based algorithm, in following study we developed another urban region detection algorithm based on Gabor features. We extracted local features in the given test image using Gabor filters. We use these local features in forming a spatial voting matrix. Then by using an optimum decision making approach, we were able to detect the urban region in given image. Comparing with SIFT based algorithm, we concluded that new urban region detection method was fairly fast and reliable. The computation time of the Gabor feature based algorithm is also very impressive. As a further step, we developed a Gabor feature based algorithm to detect urban region development. We used these local features in forming a spatial voting matrix. Then, we defined five different land development measures on it. After extensive testings, we observed that our novel land development measures have similar or better performances compared to previous land development measures. Our method did not need to register two images perfectly. This is the main contribution to the related literature. Besides, the computational cost of our novel land development measures is fairly low compared to previous methods.

In Chapter 3, we introduced novel methods to detect buildings in very high resolution satellite and aerial images. First, we started with developing our SIFT based urban region detection approach that we introduced in Section 2.2. We introduced a novel graph cut method to detect separate buildings. Unfortunately, SIFT based method needs template buildings and timing requirements are fairly high. To overcome drawbacks of SIFT based building detection algorithm, we introduced three local feature extraction methods (Harris, GMSR, and Gabor filtering) in Section 3.3. Using these descriptor vectors and a probabilistic approach we detected building centers. However, performance of system decreased slightly comparing with SIFT based algorithm, computation time requirements of the Gabor feature based system was impressive. Proposed system was very useful for real-time applications. We also fused three local features. We observed that our method is able to detect most of the buildings (having different size, shape and intensity values) in a correct manner on both aerial and satellite images. However, for aerial images the false alarm rate is fairly high. To decrease this false alarm rate, we proposed a more robust feature in Section 3.4 using steerable filters. We detected buildings regarding curvature of these features. Using this structural information increased the robustness of our algorithm. Hence, we obtained the lowest false alarm rate in aerial image data set using steerable filtering based features.

In order to increase the detection capability in aerial images, we developed a novel building detection system using color invariant features in Section 3.5. We used color invariant features to extract red-rooftops and shadows. We verify the appearance of buildings (having red and non-red rooftops) using shadow information. As a final step, we developed a novel method (box-fitting) to determine building shapes. There are many shape detection algorithms in the literature (such as gradient vector flow, and level-set algorithm), but computation times of these algorithms are very high. Contribution of our box-fitting approach was low computation time requirements when approximate shape detection is needed (even for remotely sensed images including high number of buildings). In Section 3.6, we used the same system to detect damaged buildings. We defined a measure to estimate damage degree using rooftop and shadow segments for each building. The most important contribution of this algorithm is that it worked without pre-knowledge about study region. Finally, in Section 3.7 we compared proposed building detection systems in terms of performance, timing requirements, and complexity.

In Chapter 4, we proposed two methods to detect road segments automatically on panchromatic satellite and color aerial images respectively. First, in Section 4.2 we proposed a novel approach to detect roads in panchromatic Ikonos satellite images. In order to detect road segments in a robust manner, we used structural features extracted using a set of steerable filters. We used the idea of perceptual grouping laws to group detected features. After detecting feature couples (two mutual edge structures of a road segment), we detected road centers. In order to recover missing road segments, we used a tracking approach. Our road segment detection performance is satisfying. Only, straight and long edges of a terrain (like edges of agricultural fields) can cause false alarms. These false alarms can be eliminated if multispectral information is used. When using structural features, we had too many false alarms in aerial images. The main cause of this problem is the spatial resolution of these images. Thus, in Section 4.3 we introduced a novel algorithm to detect asphalt road segments using only color information. For this purpose, we used training asphalt road patches and detect similar regions in given test image. In order to find similar regions, we proposed a novel approach based on color features extracted from CIELab color space. The performance of the proposed algorithm may be increased by fusing the color and structural features.

In our all object detection systems, we could not prevent detecting some false objects if they have straight and noteworthy edges similar to building or road edges. Especially, we have slightly higher false alarms on aerial test image data set. Since these images have very high spatial

resolution, edges and corners of non-interested objects are also visible. These redundant edge and corners lead false local feature extraction. As a result, false detection increase in these images. Unfortunately, if we try to eliminate these redundant edges and corners with smoothing, building edge and corners are also smoothed. Therefore, we lose information by smoothing, and our true detection performance also decrease. In the future, more advanced pre-processing methods can be developed for very high resolution aerial images. In each object detection method, false alarms can also be reduced by fusing features extracted in grayscale with color features. In building detection, however we obtained promising results on our test images, our methods may not discriminate adjoint buildings. In SIFT based building detection method, it is not possible to cut graph edges between buildings if intensity of buildings are very similar, and if they are very close to each other. In Gabor feature based and steerable feature based methods, probabilities of adjoint buildings will be summed. Therefore, in our each building detection method, adjoint buildings will be detected as one single building. If robust shape features can be extracted, they can help to discriminate very close objects and also to verify buildings. This problem will be investigated in our future studies.

Throughout this dissertation, we used very high resolution panchromatic satellite and aerial images (color and grayscale) to test each method. Urban region, buildings, and roads in these images have diverse characteristics. The proposed urban region, building, and road detection algorithms work fairly well. Our novel methods may be used to generate land maps, detect houses, and extract road networks automatically. Using proposed approaches, it is also possible to detect approximate shapes of the buildings, damaged buildings, and measure of development in time. Therefore, we believe that the proposed methods in this study will be of great use for urban monitoring, city planning, land mapping, disaster management, and map updating purposes.

REFERENCES

1. Colwell, R. N., *History and place of photographic interpretation*, American Society for Photogrammetry and Remote Sensing, second edn., 1997.

2. Ünsalan, C., *Multispectral satellite image understanding*, Ph.D. thesis, The Ohio State University, 2003.

3. Karathanassi, V., C. Iossifidis and D. Rokos, "*A texture-based classification method for classifying built areas according to their density*", *International Journal of Remote Sensing*, Vol. 21, No. 9, pp. 1807–1823, 2000.

4. Benediktsson, J. A., M. Pesaresi and K. Arnason, "*Classification and feature extraction for remote sensing images from urban areas based on morphological transformations*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 41, No. 9, pp. 1940–1949, 2003.

5. Ünsalan, C. and K. L. Boyer, "*A system to detect houses and residential street networks in multispectral satellite images*", *Computer Vision and Image Understanding*, Vol. 98, pp. 432–461, 2005.

6. Ünsalan, C. and K. L. Boyer, "*Classifying land development in high resolution panchromatic satellite images using straight line statistics*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 42, No. 4, pp. 907–919, 2004.

7. Ünsalan, C. and K. L. Boyer, "*Linearized vegetation indices based on a formal statistical framework*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 42, pp. 1575–1585, 2004.

8. Ünsalan, C. and K. Boyer, "*Classifying land development in high resolution satellite imagery using hybrid structural - multispectral features*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 42, No. 12, pp. 2840–2850, 2004.

9. Ünsalan, C. and K. L. Boyer, "*A theoretical and experimental investigation of graph theoretical measures for land development in satellite imagery*", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 4, pp. 575–589, 2005.

10. Fonte, L. M., S. Gautama, W. Philips and W. Goeman, "*Evaluating corner detectors for the extraction of man made structures in urban areas*", *IEEE International Geoscience and Remote Sensing Symposium*, pp. 237–240, 2005.

11. Bhagavathy, S. and B. S. Manjunath, "*Modeling and detection of geospatial objects using texture motifs*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 12, pp. 3706–3715, 2006.

12. Bruzzone, L. and L. Carlin, "*A multilevel context-based system for classification of very high spatial resolution images*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 9, pp. 2587–2600, 2006.

13. Fauvel, M., J. Chanussot and J. A. Benediktsson, "*Decision fusion for the classification of urban remote sensing images*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 10, pp. 2828–2838, 2006.

14. Zhong, P. and R. Wang, "*A multiple conditional random fields ensemble model for urban area detection in remote sensing optical images*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 45, No. 12, pp. 3978–3988, 2007.

15. Lorette, A., X. Descombes, and J. Zerubia, "*Texture analysis through a markovian modelling and fuzzy classification: application to urban area extraction from satellite images*", *International Journal of Computer Vision*, Vol. 36, No. 3, pp. 221–236, 2000.

16. Aksoy, S. and E. Dogrusöz, "*Modeling urban structures using graph-based spatial patterns*", *IEEE International Geoscience and Remote Sensing Symposium*, pp. 4826 – 4829, 2007.

17. Gamba, P., F. D. Acqua, G. Lisini and G. Trianni, "*Improved VHR urban area mapping exploiting object boundaries*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 45, No. 8, pp. 2676–2682, 2007.

18. Yang, Y. and S. Newsam, "*Comparing SIFT descriptors and Gabor texture features for classification of remotely sensed imagery*", *IEEE International Conference on Image Processing*, 2008.

19. Laha, A., N. R. Pal and J. Das, "*Land cover classification using fuzzy rules and aggregation of contextual information through evidence theory*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 6, pp. 1633–1641, 2006.

20. Lee, S. and R. G. Lathrop, "*Subpixel analysis of Landsat ETM+ using self-organizing map (SOM) neural networks for urban land cover characterization*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 6, pp. 1642–1654, 2006.

21. Bruzzone, L., M. Chi, and M. Marconcini, "*A novel transductive SVM for semisupervised classification of remote-sensing images*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 11, pp. 3363–3373, 2006.

22. Mari, J. M., L. Bruzzone and G. C. Valls, "*A support vector domain description approach to supervised classification of remote sensing images*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 45, No. 8, pp. 2683–2692, 2007.

23. Hernandez, C. S., D. S. Boyd and G. M. Foody, "*One-class classification for mapping a specific land-cover class: SVDD classification of Fenland*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 45, No. 4, pp. 1061–1073, 2007.

24. Lee, J. and O. K. Ersoy, "*Consensual and hierarchical classification of remotely sensed multispectral images*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 45, No. 9, pp. 2953–2963, 2007.

25. Kosugi, Y., M. Sakamoto, M. Fukunishi, W. Lu, T. Doihara and S. Kakumoto, "*Urban change detection related to earthquakes using an adaptive nonlinear mapping of high resolution images*", *IEEE Geoscience and Remote Sensing Letters*, Vol. 1, pp. 152–156, 2004.

26. Mura, M. D., J. A. Benediktsson, F. Bovolo and L. Bruzzone, "*An unsupervised technique based on morphological filters for change detection in very high resolution images*", *IEEE Geoscience and Remote Sensing Letters*, Vol. 5, pp. 433–437, 2008.

27. Radke, R. J., S. Andra, O. Al-Kofahi and B. Roysam, "*Image change detection algorithms: A systematic survey*", *IEEE Transactions on Image Processing*, Vol. 14, No. 3, pp. 294–307, 2005.

28. Li, W., X. Li, Y. Wu and Z. Hu, "*A novel framework for urban change detection using VHR satellite images*", *International Conference on Pattern Recognition*, Vol. 2, pp. 312–315, 2006.

29. Tang, F. and V. Prinet, "*Computing invariants for structural change detection in urban areas*", *Urban Remote Sensing Joint Event*, pp. 1–6, 2007.

30. Ünsalan, C., "*Measuring land development in urban regions using graph theoretical and conditional statistical features*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 45, No. 12, pp. 3989–3999, 2007.

31. Lowe, D. G., "*Distinctive image features from scale invariant keypoints*", *International Journal of Computer Vision*, Vol. 60, No. 2, pp. 91–110, 2004.

32. Suga, A., K. Fukuda, T. Takiguchi and Y. Ariki, "*Object recognition and segmentation using SIFT and graph cuts*", *19th International Conference on Pattern Recognition*, pp. 1–4, 2008.

33. Hu, X., Y. Tang and Z. Zhang, "*Video object matching based on SIFT algorithm*", *International Conference on Neural Networks and Signal Processing*, pp. 412–415, 2008.

34. Stein, A. and M. Hebert, "*Incorporating background invariance into feature-based object recognition*", *Seventh IEEE Workshops on Application of Computer Vision*, Vol. 1, pp. 37–44, 2005.

35. Fritz, G., C. Seifert, M. Kumar and L. Paletta, "*Building detection from mobile imagery using informative SIFT descriptors*", *14th Scandinavian Conference on Image Analysis*, pp. 629–638, 2005.

36. Mikolojczyk, K. and C. Schmid, "*A performance evaluation of local descriptors*", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 10, pp. 1615–1630, 2005.

37. Tomasi, C. and R. Manduchi, "*Bilateral filtering for gray and color images*", *Proceedings of the International Conference on Computer Vision*, pp. 839–846, 1998.

38. Elad, M., "*On the origin of bilateral filter and ways to improve it*", *IEEE Transactions on*

Image Processing, Vol. 11, No. 10, pp. 1141–1151, 2002.

39. Sonka, M., V. Hlavac and R. Boyle, *Image processing, analysis and machine vision*, PWS Publications, Pacific Grove, CA, second edn., 1999.

40. Kyrki, V., J. K. Kamarainen and H. Kalviainen, "*Simple Gabor feature space for invariant object recognition*", *Pattern Recognition Letters*, Vol. 25, No. 3, pp. 311–318, 2004.

41. Jain, A. K., N. K. Ratha and S. Lakshmanan, "*Object detection using Gabor filters*", *Pattern Recognition*, Vol. 30, No. 2, pp. 295–309, 1997.

42. Weber, D. M. and D. P. Casasent, "*Quadratic Gabor filters for object detection*", *IEEE Transactions on Image Processing*, Vol. 10, No. 2, pp. 218–230, 2001.

43. Kumar, A. and G. K. H. Pang, "*Defect detection in textured materials using Gabor filters*", *IEEE Transactions on Industry Applications*, Vol. 38, No. 2, pp. 425–440, 2002.

44. Vetterli, M. and J. Kovacevic, *Wavelets and subband coding*, Prentice Hall, first edn., 1995.

45. Otsu, N., "*A threshold selection method from gray-level histograms*", *IEEE Transactions on System, Man, and Cybernetics*, Vol. 9, No. 1, pp. 62–66, 1979.

46. Paris, S. and F. Durand, "*A fast approximation of the bilateral filter using a signal processing approach*", *International Journal of Computer Vision*, Vol. 81, No. 1, pp. 24–52, 2009.

47. Mayer, H., "*Automatic object extraction from aerial imagery - a survey focusing on buildings*", *Computer Vision and Image Understanding*, Vol. 74, pp. 138–149, 1999.

48. Levitt, S. and F. Aghdasi, "*Fuzzy representation and grouping in building detection*", *International Conference on Image Processing*, Vol. 3, pp. 324–327, 2000.

49. Kim, T. and J. P. Muller, "*Development of a graph-based approach for building detection*", *Image and Vision Computing*, Vol. 17, No. 1, pp. 3–14, 1999.

50. Segl, K. and H. Kaufmann, "*Detecting small objects from high-resolution panchromatic satellite imagery based on supervised image segmentation*", *IEEE Transactions on Geo-*

science and Remote Sensing, Vol. 39, No. 9, pp. 2080–2083, 2001.

51. Mayunga, S. D., Y. Zhang and D. J. Coleman, "*Semi-automatic building extraction utilizing Quickbird imagery*", *ISPRS City Models, Roads and Traffic Workshop*, pp. 131–136, 2005.

52. Peng, J., D. Zhang and Y. Liu, "*An improved snake model for building detection from urban aerial images*", *Pattern Recognition Letters*, Vol. 26, pp. 587–595, 2005.

53. Huang, X., L. Zhang and P. Li, "*An adaptive multiscale information fusion approach for feature extraction and classification of Ikonos multispectral imagery over urban areas*", *IEEE Geoscience and Remote Sensing Letters*, Vol. 4, No. 4, pp. 654–658, 2007.

54. Molinier, M., J. Laaksonen and T. Hame, "*Detecting man-made structures and changes in satellite imagery with a content-based information retrieaval system built on self-organizing maps*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 45, No. 4, pp. 861–874, 2007.

55. Wei, W. and Y. Xin, "*Feature extraction for man-made objects segmentation in aerial images*", *Machine Vision and Applications*, Vol. 19, pp. 57–64, 2008.

56. Katartzis, A. and H. Sahli, "*A stochastic framework for the identification of building rooftops using a single remote sensing image*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 46, No. 1, pp. 259–271, 2008.

57. Karantzalos, K. and N. Paragios, "*Recognition-driven two-dimensional competing priors toward automatic and accurate building detection*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 47, No. 1, pp. 133–144, 2009.

58. Akçay, H. G. and S. Aksoy, "*Automatic detection of geospatial objects using multiple hierarchical segmentations*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 46, No. 7, pp. 2097–2111, 2008.

59. Idrissa, M., V. Lacroix, A. Hincq, H. Bruynseels and O. Swartenbroekx, "*SPOT5 images for urbanization detection*", *Proceedings of Advanced Concepts for Intelligent Vision Systems*, 2004.

60. Irvin, R. B. and D. M. McKeown, "*Methods for exploiting the relationship between buildings and their shadows in aerial imagery*", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 19, No. 1, pp. 1564–1575, 1989.

61. Zimmermann, P., "*A new framework for automatic building detection analyzing multiple cue data*", *International Archives of Photogrammetry and Remote Sensing*, Vol. 33, pp. 1063–1070, 2000.

62. Tsai, V. J. D., "*A comparative study on shadow compensation of color aerial images in invariant color models*", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 6, pp. 1661–1671, 2006.

63. Vu, T., M. Matsouka and F. Yamazaki, "*Shadow analysis in assisting damage detection due to earthquake from quickbird imagery*", *10th International Society for Photogrammetry and Remote Sensing Congress*, pp. 607–611, 2004.

64. Chen, Z. and T. C. Hutchinson, "*A probabilistic classification framework for urban structural damage estimation using satellite images*", *Urban Remote Sensing Joint Event*, pp. 1–7, 2007.

65. Wei, L. and V. Prinet, "*Building detection from high-resolution satellite image using probability model*", *IEEE International Geoscience and Remote Sensing Symposium*, pp. 3888–3891, 2005.

66. Jaynes, C. O., F. Stolle and R. T. Collins, "*Task driven perceptual organization for extraction of rooftop polygons*", *Proceedings of the Second IEEE Workshop on Applications of Computer Vision*, pp. 152–159, 1994.

67. Mordohai, P. and G. Medioni, "*Stereo using monocular cues within the tensor voting framework*", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, No. 6, pp. 968–982, 2006.

68. Huertas, A. and R. Nevatia, "*Detecting buildings in aerial images*", *Computer Vision, Graphics and Image Processing*, Vol. 41, pp. 131–152, 1988.

69. Chellappa, R. and S. Krishnamarchi, "*Delienating buildings by grouping lines with MRF*",

IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 5, No. 1, pp. 164–168, 1996.

70. Neuenschwander, W., P. Fua, G. Székely and O. Kübler, *Automatic extraction of man-made objects from aerial and space images*, Birkhäuser Verlang, 1995.

71. Ruther, H., H. M. Martine and E. G. Mtalo, "*Application of snakes and dynamic programming optimization technique in modelling of buildings in informal settlement areas*", *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 56, No. 4, pp. 269–282, 2002.

72. Ünsalan, C., "*Gradient magnitude based support regions in structural land use classification*", *IEEE Geoscience and Remote Sensing Letters*, Vol. 3, No. 4, pp. 546–550, 2006.

73. Fonte, L. M., S. Gautama, W. Philips and W. Goeman, "*Evaluating corner detectors for the extraction of man made structures in urban areas*", *IEEE International Geoscience and Remote Sensing Symposium*, pp. 237–240, 2005.

74. Schmid, C., R. Mohr and C. Bauckhage, "*Evaluation of interest point detectors*", *International Journal of Computer Vision*, Vol. 37, No. 2, pp. 151–172, 2000.

75. Harris, C. and M. Stephens, "*A combined corner and edge detector*", *Proceedings of the Fourth Alvey Vision Conference*, pp. 147–151, 1988.

76. Silverman, B. W., *Density estimation for statistics and data analysis*, Chapman Hall, first edn., 1986.

77. Freeman, W. and E. Adelson, "*The design and use of steerable filters*", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, pp. 891–906, 1991.

78. Orrite, C., A. Roy and A. Alcolea, "*Surface segmentation based on perceptual grouping*", *International Conference on Image Analysis and Processing*, pp. 328–333, 1999.

79. Gevers, T. and A. W. M. Smeulders, "*PictoSeek: combining color and shape invariant features for image retrieval*", *IEEE Transactions on Image Processing*, pp. 102–119, 2000.

80. Zhang, K., J. Yan and S. C. Chen, "*Automatic construction of building footprints from air-*

borne LIDAR data", *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 9, pp. 2523–2533, 2006.

81. Canny, J., "*A computational approach to edge detection*", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 6, pp. 679–698, 1986.

82. Udomhunsakul, S., "*Semi-automatic road detection from satellite imagery*", *International Conference on Image Processing*, Vol. 3, pp. 1723–1726, 2004.

83. Ameri, F., A. M. Mobarraki and M. J. Valadan Zoej, "*Semi-automatic extraction of different shaped road centerlines from MS and pan-sharpend Ikonos images*", *XXIth Congress of The International Society for Photogrammetry and Remote Sensing*, pp. 621–626, 2008.

84. Dell'Acqua, F., P. Gamba and G. Lisini, "*A semi-automatic high resolution SAR data interpretation procedure*", *International Conference on Photogrammetric Image Analysis*, pp. 19–24, 2007.

85. Yang, J. and R. S. Wang, "*Classified road detection from satellite images based on perceptual organization*", *International Journal of Remote Sensing*, Vol. 28, No. 20, pp. 4653–4669, 2007.

86. Ma, H., Q. Qin, S. Du, L. Wang and C. Jin, "*Road extraction from ETM panchromatic image based on dual-edge following*", *IEEE International Geoscience and Remote Sensing Symposium*, pp. 460–463, 2007.

87. Rianto, Y. and S. Kondo, "*Detection of roads from satellite image using optimal search*", *IEEE Asia-Pacific Conference on Circuits and Systems*, pp. 587–590, 1998.

88. Baumgartner, A., C. Steger, H. Mayer and W. Eckstein, "*Multi-resolution, semantic objects, and context for road extraction*", *In Semantic Modeling for the Acquisition of Topographic Information from Images and Maps*, pp. 140–156, Birkhauser Verlag, 1997.

89. Montesinos, P. and L. Alquier, "*Perceptual organization of thin networks with active contour functions applied to medial and aerial images*", *13th International Conference on Pattern Recognition*, Vol. 1, pp. 647–651, 1996.

90. Sarkar, S. 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*, Vol. 23, No. 2, pp. 382–399, 1993.

91. Gonzalez, R. C. and R. E. Woods, *Digital Image Processing*, Prentice Hall, New Jersey, second edn., 2002.

92. Fairchild, M., *Color Appearance Models*, Addison-Wesley, 1998.

93. Paschos, G., "*Perceptually uniform color spaces for color texture analysis: an empirical evaluation*", *IEEE Transactions on Image Processing*, Vol. 10, pp. 932–937, 2001.

94. Haoting, L., G. Jiang and L. Wang, "*Multiple objects tracking based on snake model and selective attention mechanism*", *International Journal of Information Technology*, Vol. 12, No. 2, pp. 76–86, 2006.

95. Moya, M., M. Koch and L. Hostetler, "*One class classifier networks for target recognition applications*", *World Congress on Neural Networks for Target Recognition Applications*, pp. 797–801, 1993.

96. Erçil, A. and B. Büke, "*One class classification using implicit polynomial surface fitting*", *16th International Conference on Pattern Recognition*, Vol. 2, pp. 152–155, 2002.

97. Duda, R. O., P. E. Hart and D. G. Stork, *Pattern Classification*, Wiley-Interscience Publication, second edn., 2000.