AN OBJECT RECOGNITION FRAMEWORK USING CONTEXTUAL INTERACTIONS AMONG OBJECTS

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Fırat Kalaycılar

August, 2009

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Selim Aksoy(Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Fatoş Tünay Yarman-Vural

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Pınar Duygulu-Şahin

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet B. Baray
Director of the Institute

ABSTRACT

AN OBJECT RECOGNITION FRAMEWORK USING CONTEXTUAL INTERACTIONS AMONG OBJECTS

Fırat Kalaycılar
M.S. in Computer Engineering
Supervisor: Assist. Prof. Dr. Selim Aksoy
August, 2009

Object recognition is one of the fundamental tasks in computer vision. The main endeavor in object recognition research is to devise techniques that make computers understand what they see as precise as human beings. The state of the art recognition methods utilize low-level image features (color, texture, etc.), interest points/regions, filter responses, etc. to find and identify objects in the scene. Although these work well for specific object classes, the results are not satisfactory enough to accept these techniques as universal solutions. Thus, the current trend is to make use of the context embedded in the scene. Context defines the rules for object - object and object - scene interactions. A scene configuration generated by some object recognizers can sometimes be inconsistent with the scene context. For example, observing a car in a kitchen is not likely in terms of the kitchen context. In this case, knowledge of kitchen can be used to correct this inconsistent recognition.

Motivated by the benefits of contextual information, we introduce an object recognition framework that utilizes contextual interactions between individually detected objects to improve the overall recognition performance. Our first contribution arises in the object detector design. We define three methods for object detection. Two of these methods, shape based and pixel classification based object detection, mainly use the techniques presented in the literature. However, we also describe another method called surface orientation based object detection. The goal of this novel detection technique is to find objects whose shape, color and texture features are not discriminative while their surface orientations (horizontality or verticality) are consistent across different instances. Wall, table top, and road are typical examples for such objects. The second contribution is a probabilistic contextual interaction model for objects based on their spatial relationships. In order to represent the spatial relationships between objects, we propose three features that encode the relative position/location, scale and orientation of a given object pair. Using these features and our object interaction likelihood model, we achieve to encode the semantic, spatial, and pose context of a scene concurrently. Our third main contribution is a contextual agreement maximization framework that assigns final labels to the detected objects by maximizing a scene probability function that is defined jointly using both the individual object labels and their pairwise contextual interactions. The most consistent scene configuration is obtained by solving the maximization problem using linear optimization.

We performed experiments on the LabelMe [27] and Bilkent data sets by both utilizing and not utilizing the scene type (indoor or outdoor) information. While the average F2 score increased from 0.09 to 0.20 without the scene type assumption, it increased from 0.17 to 0.25 when the scene type is known on the LabelMe dataset. The results are similar for the experiments performed on the Bilkent data set. F2 score increased from 0.16 to 0.36 when the scene type information is not available and it increased from 0.31 to 0.44 when this additional information is used. It is clear that the incorporation of the contextual interactions improves the overall recognition performance.

Keywords: Context, object recognition, spatial relationships.

ÖZET

NESNELER ARASINDAKİ BAĞLAMSAL ETKİLEŞİMLERİ KULLANAN BİR NESNE TANIMA ÇERÇEVESİ

Fırat Kalaycılar
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Yard. Doç. Dr. Selim Aksoy
Ağustos, 2009

Nesne tanıma, bilgisayarlı görme alanının en temel problemlerinden biridir. Bilgisayarlar gördüklerini insanlar gibi anlayabilsin diye teknikler geliştirmek nesne tanıma araştırmalarındaki ana uğraştır. Bir sahnedeki nesneleri bulmak ve tanımlayabilmek için en çok kullanılan yöntemlerde, alt-düzey görüntü öznitelikleri (renk, doku, vb.), ilgi noktaları/bölgeleri, süzgeç tepkileri, vb. özelliklerden yararlanılmaktadır. Bunlar belirli nesne sınıfları için düzgün çalışsa da, genel bir çözüm olmaktan uzaktırlar. Bu yüzden, sahne bağlamını kullanmak güncel bir eğilim halini almıştır. Bağlam nesneler arası ve nesne - sahne arası ilişkilerin kurallarını belirlemektedir. Nesne tanıyıcıların ortaya çıkardığı sahne düzenleşimleri bazı durumlarda sahne bağlamıyla örtüşmemektedir. Örneğin, bir mutfak ortamında araba görülmesi mutfak bağlamı açısından pek olası değildir. Bu durumda, mekanın bir mutfak olduğunu bilmek bu tür çelişkili tanımlamaları engellemekte kullanılabilir.

Bağlamsal bilginin getirdiği faydaları hesaba katarak, bu tezde, nesne tanıma başarımını arttırmak için tek tek sezilmiş nesneler arasındaki bağlamsal etkileşimlerden yararlanan bir nesne tanıma çerçevesi anlatılmaktadır. İlk katkımız nesne sezicilerin tasarımında görülmektedir. Çerçevemizde üç farklı nesne sezim yöntemi tanımlanmıştır. Bunlardan ikisi, şekil bazlı ve piksel sınıflandırması bazlı nesne sezicilerdir ve tasarımlarında genel olarak varolan yöntemlerden yararlanılmaktadır. Bunlardan başka, yüzey doğrultusu bazlı nesne sezici isimli üçüncü bir yöntem geliştirilmiştir. Bu yeni nesne sezim yöntemindeki ana amaç, şekil, renk ve doku özellikleri ayırt edici olmasa da yüzey doğrultuları (diklik ya da yataylık durumları) tutarlı olan nesnelerin sezilebilmesini sağlamaktır. Duvar, masa üstü, yol, vb. nesneler bu gruba dahil edilmektedir. İkinci katkımız, nesneler arasındaki uzamsal ilişkilere dayanan bağlamsal etkileşim modelidir. Nesneler arasındaki uzamsal ilişkileri göstermek için göreli konum, ölçek ve doğrultu bilgilerini içeren üç tane öznitelik tanımlanmıştır. Bu öznitelikleri ve nesne etkileşim olurluğu modelini kullanarak sahnenin anlamsal, uzamsal ve duruş bağlamları aynı anda ifade edilebilmektedir. Üçüncü ana katkımız, bireysel nesne etiketlerine ve nesne ikilileri arasındaki etkileşimlere bağlı olan sahne olasılık fonksiyonunun enbüyütülerek, nesnelerin en son etiketlerinin atanmasıdır. En tutarlı sahne düzenleşimini bulmak için bu enbüyütme problemi, doğrusal eniyileme kullanılarak çözülmüştür.

LabelMe [27] ve Bilkent veri kümelerinde, hem sahne türünü (iç mekan ya da dış mekan) hesaba katarak hem de katmayarak deneyler gerçekleştirilmiştir. LabelMe veri kümesinde sahne türü bilgisi kullanılmadığında F2 başarı ölçütü 0.09'dan 0.20'ye yükselmiştir. Sahne türü bilgisinden yararlanıldığında F2 ölçütü 0.17'den 0.25 değerine ulaşmıştır. Benzer başarım artışları Bilkent veri kümesinde gerçekleştirilen deneylerde de görülmüştür. Sahne türü hesaba katılmadığında F2 ölçütü 0.16'dan 0.36'ya yükselirken, sahne türü dikkate alındığında ölçüt, 0.31 değerinden 0.44 değerine yükselmiştir. Bu deneyler sonucunda, bağlamsal etkileşimlerin nesne tanıma başarımına olumlu bir etkisi olduğu gösterilmiştir.

Anahtar sözcükler: Bağlam, nesne tanıma, uzamsal ilişkiler.

Acknowledgement

I would like to express my deep thanks to my advisor, Selim Aksoy, for his guidance and support throughout this work. It has been a valuable experience for me to work with him and get benefit from his vision and knowledge in every step of my research.

I am very thankful to Fatoş Tünay Yarman-Vural for inspiring me to choose computer vision as my research topic. If I had not met her at METU, I might have missed the chance to work on computer vision which now holds a great interest for me. In addition, I would like to thank her for the suggestions on improving this work.

I also would like to render thanks to Pınar Duygulu-Şahin for reviewing this thesis and providing useful feedback about the work presented here.

Certainly, I am grateful to my whole family for always being by my side. Without their endless support and caring, completion of this thesis would not be possible.

Aslı, Bahadır, Çaglar, Daniya, Gökhan, Nazlı, Onur, Sare, Selen, Sıtar, Zeki and all other people I met at Bilkent made the last two years of my life very special and unforgettable. I cannot deny that their nice friendship and moral support helped me much in hard times.

Besides, I would like to express my pleasure on being part of Bilkent University and RETINA Group where I could always feel warmth and sincerity.

Finally, I would like to thank TÜBİTAK BİDEB (The Scientific and Technological Research Council of Turkey) for their financial support.

To my family . . .

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview and Related Work

Object recognition is one of the fundamental tasks in computer vision. It is defined as the identification of objects that are present in the scene. For a human being, recognition of objects is a straightforward action. This ability works well for the identification of objects in both videos and still images. The success in this task is not affected drastically even when the scene properties alter. For example, human vision system is nearly invariant to the changes in *illumination* conditions. Therefore, recognizing an object in a darker scene is not very much different than doing this in a brighter environment as long as the level of illumination is reasonable. Similarly, the *occlusion* of objects does not influence the recognition performance in a serious way. When the *background clutter* is considered, again human beings are good at disguishing an object from a complex background. Likewise, the different *poses* or *visual diversity* of objects do not have a significant effect on the recognition quality of the human vision system.

However, when a computer is programmed to mimic this seemingly simple ability of recognizing objects, unfortunately, the results are usually not satisfactory. Factors like illumination, occlusion conditions, background clutter, variety in pose and appearance are important challenges for computer vision. Thus, for decades, vision scientists have been trying to develop algorithms to make computers understand what they *see* as precise as human beings.

There are different lines of research for the object recognition task. One of them is to decide if a member of an object class is present or absent in a given image regardless of its location. In [5, 8, 19], examples of this approach are described. The methods work well for images that contains a single object of interest and a relatively simple background. In this approach, first of all, some features (color, texture, etc.) and/or interest points (SIFT [21], Harris interest points [22] and etc.) are extracted. Then, decision on the existence of the object is made using the features and interest points.

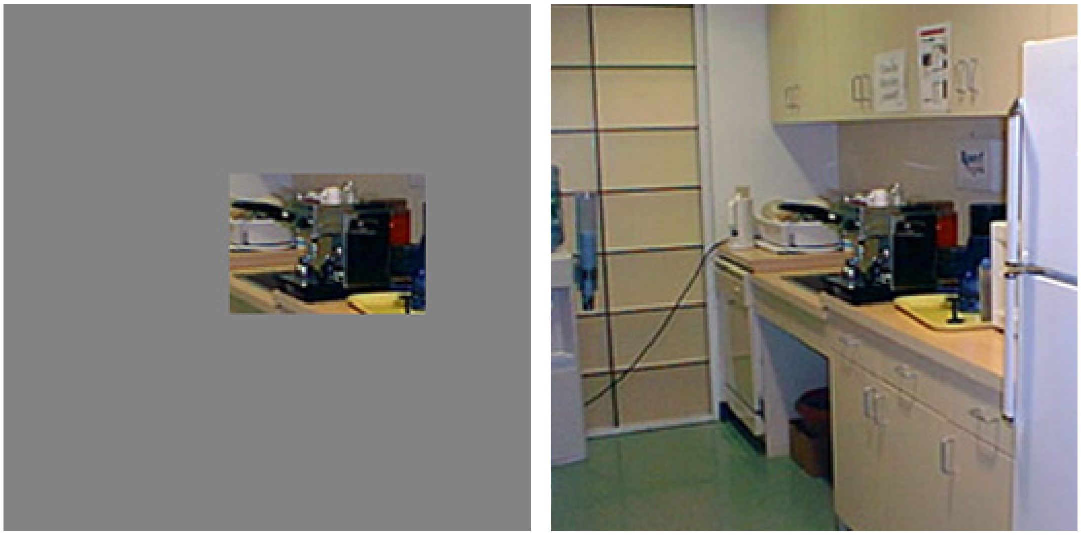
The second line of research includes region-based methods that use segmentation to determine the boundaries of each object, and identify the objects by classifying these segments. An example work using this approach is explained in [3]. Although this method seems intuitive, it is often not possible to have a perfect segmentation. The most of the segmentation algorithms work well for scenes where objects have definite texture and color features. However, most of the objects are composed of parts with different properties so algorithms often segment these objects into several useless parts. To avoid this problem, multiple segmentation approaches have been used [13, 24, 26]. The idea behind multiple segmentations is that using different segmentation algorithms or running the same algorithms with different parameters may yield different segmentations of the image. Using the sets of pixels that are grouped into the same segments for different settings as consistency information, these methods try to decide on the most stable segmentation and use it for object recognition. Unfortunately, this approach usually works well when the number of objects in the scene is small.

The third line of research is direct detection where the input image is divided into overlapping tiles, and for each tile, the classifier decides if there is an object of a particular class or not [11, 28]. Alternatively, the detectors can make use of feature responses to determine the probable location of a target object [32]. After deciding on the tiles or the locations of target objects, detectors report the bounding boxes and the associated probabilities of existence. Another direct detection method is part-based object recognition. In this method, first of all, discriminative parts of an object are detected. Then, using the individual parts and the part-based object model (relationship between the parts), the entire object can be captured. Example work using part-based object recognition can be found in [9, 10, 12].

None of these object recognition methods provides a universal methodology to solve the object recognition problem. The reason is that they do not make use of any high level information. They analyze the low level image features, filter responses, interest points, segmentations, etc. They do not take the structure and the constraints of the scene into account during the analysis. Actually, there is a rich source of high level information called *context* that can be used to enhance the recognition quality. Context refers to the rules and constraints of the object-object and object-scene interactions. Human vision and computer vision researchers [2, 15, 23] acknowledge that these contextual rules and constraints are useful while recognizing an object.

The literature on the visual perception [23] shows that context has influences at different levels. The *semantic* context represents the meaningfulness of the co-occurrences of particular objects. While table and chair have higher tendency to be found in the same scene, observing a car and a keyboard in the same environment is not likely. The *spatial* context puts constraints on the expected positions and locations of the objects. For instance, when a person sees an object flying in the sky, he knows that it is likely to be a bird or an airplane. The reason is that his cognitive system is aware that the probability of observing a flying tree, building, car, etc. is significantly low. In addition, the *pose* context defines the tendencies in the relative poses of the objects with respect to each other. For instance, we, human beings, anticipate that a car is oriented meaningfully on the road while driving.

Figure 1.1 shows an example emphasizing the importance of the contextual interactions [33] for human vision systems. When the whole scene is not visible, it is sometimes impossible to recognize the object of interest. But, recognition becomes easier when we see an object in the context of the scene it belongs to.



(a) Object (coffee machine) without scene context.

(b) Object (coffee machine) inside the scene (kitchen) context.

Figure 1.1: Example of object (coffee machine) - scene (kitchen) interaction [33].

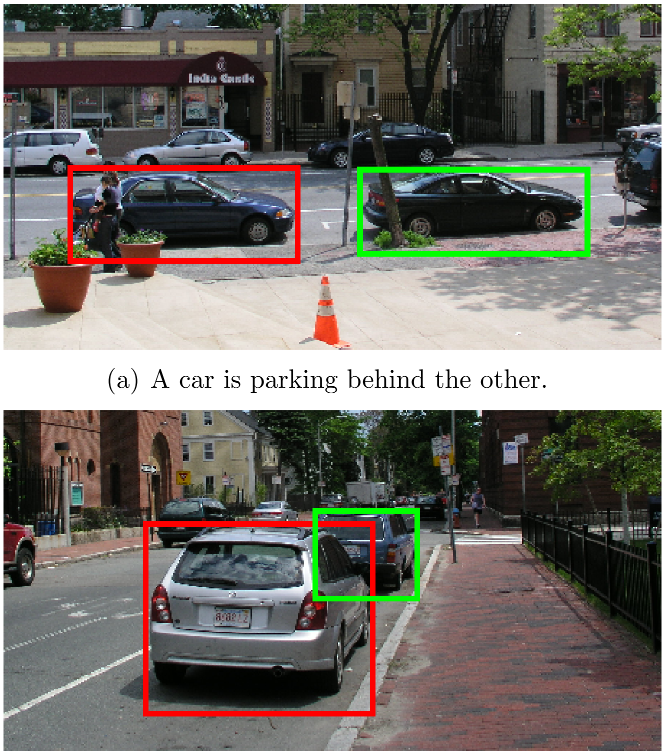
In Figure 1.1(a), only the object of interest is shown. By just looking at that image, it is very hard to determine the type of the object. But, when it is known that it is a kitchen scene, it can be easily understood that it is a coffee machine as shown in Figure 1.1(b).

Motivated by the benefits of context for human visual perception, computer vision researchers devise techniques in which this rich information is incorporated into the entire recognition process. In [31], Torralba introduced a framework that models the relationship between the context and objects using the correlation between the statistics of the low-level features extracted from the whole scene and individual objects. In another work, Torralba et al. [33] proposed a context-based vision system in which global features are used in the scene prediction. The contextual information originating from that scene provides prior information for the local detectors. The main drawback of these approaches is that the scene context is modeled in terms of low-level features that may vary from image to image.

Besides the methods that deal with the correlations between low-level features, more intuitive models also exist. In [26], context in the scene is modeled in terms of the co-occurrence likelihoods of objects (semantic context). An image is segmented into stable regions, and for each region, the class labels are sorted from the most to the least likely. By the help of the contextual model, the method manages to disambiguate each region and assigns a final class label. The main drawback of this method is its dependency on the quality of the image segmentations. The most of the natural scenes are composed of many objects of interest and a cluttered background. The state of the art algorithms often cannot segment such natural scene images meaningfully. Thus, this method can work best for the images with relatively simpler scene configurations that are appropriate for the segmentation algorithms.

There are also several methods that make use of the spatial context besides the semantic context. In [13], in addition to co-occurrences, the relative location information (spatial relationships) is also considered as contextual information. [25] also uses co-occurrences, location and relative scale as contextual interactions. The common aspect of these studies is the use of the conditional random field (CRF) framework for incorporation of the contextual information. Note that only a few number of objects as variables can be handled using CRF due to its computational complexity.

Use of pairwise relations between region/object as spatial context is common [4, 14, 18, 20, 29]. Incorporation of region/object spatial relationships such as above, below, right, left, surrounding, near, etc, into the decision process has been shown to improve the recognition performance in satellite image analysis [1]. However, the models that extract such relationships from a 2D image taken from a top view of the Earth may not be valid for generic views in natural scenes. Since these relationships are not invariant to changes in perspective projection, an actual spatial relationship defined in the real 3D world can be perceived very differently in the image space. Figure 1.2 shows such a setting. In both of the images, the car with the red marker is parked behind the car with the green marker. However, when the 2D spatial relationships of the cars are interpreted, they seem to have different relative positions. Therefore, naming the observed 2D relationships as above, below, etc. does not always make sense in generic views. In [17], we proposed a method that probabilistically infers the real world



(a) A car is parking behind the other.

(b) The same relationship from a different angle of view.

Figure 1.2: Similar object relationships in the 3D world can appear in many different configurations in 2D images.

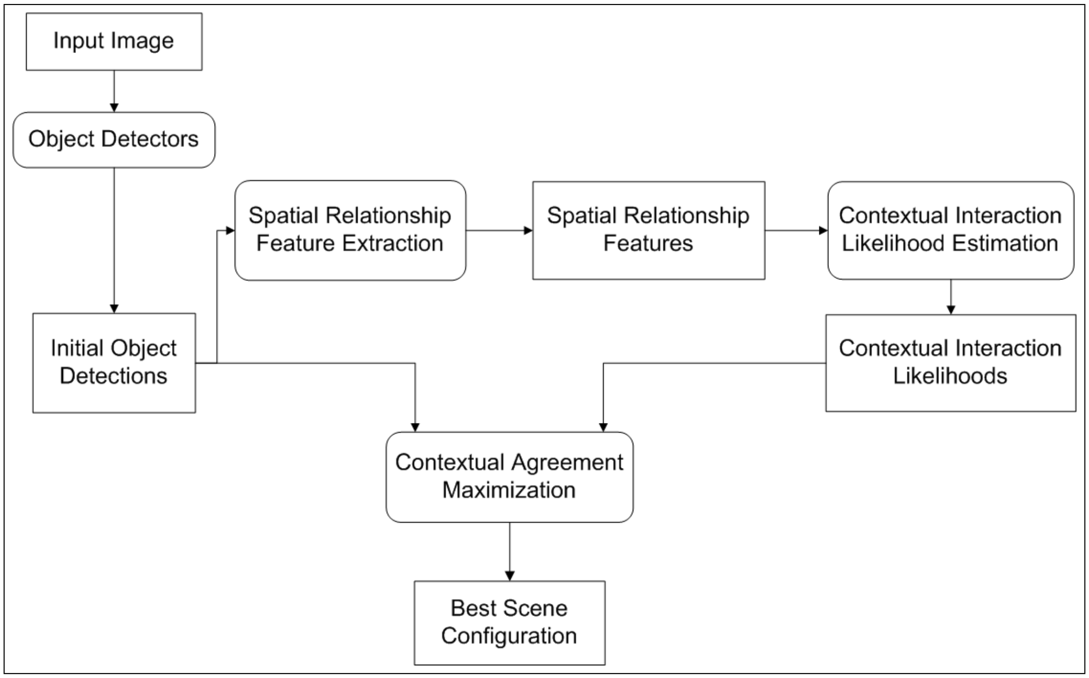

Input Image
Object Detectors
Spatial Relationship Feature Extraction
Spatial Relationship Features
Contextual Interaction Likelihood Estimation
Initial Object Detections
Contextual Interaction Likelihoods
Contextual Agreement Maximization
Best Scene Configuration

Figure 1.3: Overview of the contextual object recognition framework.

relationships (defined in 3D space) between objects and use them as spatial context. Although the results are promising, the scalability of the context model is problematic due to issues with training real world relationship models.

In this work, we introduce a contextual object recognition framework where the main goal is the determination of the best scene configuration for a still image of a natural indoor/outdoor scene with a complex background and many objects of interest. For this purpose, first, all object detectors are run on the input image. This procedure yields the initial object detections with the class membership probabilities. Then, the contextual interactions among these candidate objects are estimated. Three new spatial relationship features that encode the relative position/location, scale and orientation are extracted for each object pair and used to compute object - object contextual interaction likelihoods. These likelihoods measure the meaningfulness of the interactions (relative position, location and pose) between the objects. Finally, our framework finds the best scene configuration by maximizing the contextual agreement between the object class labels and their contextual interactions encoded in a novel scene probability function. In our framework, finding the best scene configuration corresponds to the elimination of the objects that are inconsistent with the scene context and the disambiguation of the multiple class labels possible for a single object. The overview of the framework is shown in Figure 1.3.

1.2 Summary of Contributions

Our contextual object recognition framework has three main parts: object detection, spatial relationship feature extraction, and contextual agreement maximization. We have contributions for each of these parts. First, we define three methods for object detection. Two of these methods, shape based and pixel classification based object detection, mainly use the techniques presented in the literature. However, we also describe another method called surface orientation based object detection. The goal of that detection technique is to find objects whose shape, color and texture features are not discriminative. Wall, table top, and road are typical examples for such objects. In order to recognize the instances of these object classes, we designed a novel object detector that makes use of the surface orientations. Details are given in Section 2.3.

Another contribution of this work is about spatial relationship features that are the measures of object-object contextual interactions. We define three novel features that encode the relative position, location, scale and orientation extracted from a given object pair. The oriented overlaps feature is the numerical representation of the overlaps observed in the projections of the objects in various orientations. Note that the overlap ratio without orientation is widely used to represent the object relationships. Since an overlap ratio without orientation is a scalar value, it lacks the direction information. On the other hand, our oriented overlaps feature contains rough information regarding the direction based on the orientations. By this way, it can encode the relative position of two objects. The second feature, oriented end points, is designed to represent the relative location of the objects in different orientations. It compares the end points of the objects in the projection space generated by projecting the objects to an orientation of interest. Note that both oriented overlaps and oriented end points are normalized features so that scaling the objects does not change the feature values. Finally, horizontality feature captures the relative horizontality of two objects as a cue about their support relationship. When these features are used in the same probabilistic model, inference about the likelihood of the object interactions can be made more robustly. More information about our features and their relation to object interactions are explained in Chapter 3.

The most important contribution is made in the contextual agreement maximization part of our framework. This part combines individual detections (output of the detectors) and the spatial relationships between them to obtain the best scene configuration (best choices for object class labels). For this purpose, we define a novel scene probability function whose maximization results in the configuration we seek. This function is defined jointly using individual object class labels and their pairwise spatial relationship features. In addition, it can easily be decomposed into computable probabilistic terms (object detection confidences and object interaction likelihoods). The values of the terms come from object detector outputs and the probabilistic contextual interactions models based on our spatial relationship features. Finally, our scene probability function can be easily extended so that its maximization can also lead to additional information about the scene configuration besides object class labels. Such additional information regarding the scene is the type of the relationships (on, under, beside, attached, etc.) between the objects. Complete description of this contextual agreement maximization framework is presented in Chapter 4.

1.3 Organization of the Thesis

The rest of the thesis is organized as follows. In the following chapters, details regarding our contextual object recognition framework are given. Chapter 2 deals with object detector designs we use to obtain initial objects. Next, in Chapter 3, spatial relationship features and object interaction likelihood models are explained. Chapter 4 shows how initial objects and their interactions are used together to reach the best scene configuration. Our data sets and experimental results are presented in Chapter 5. Finally, conclusions together with future work are given in Chapter 6.

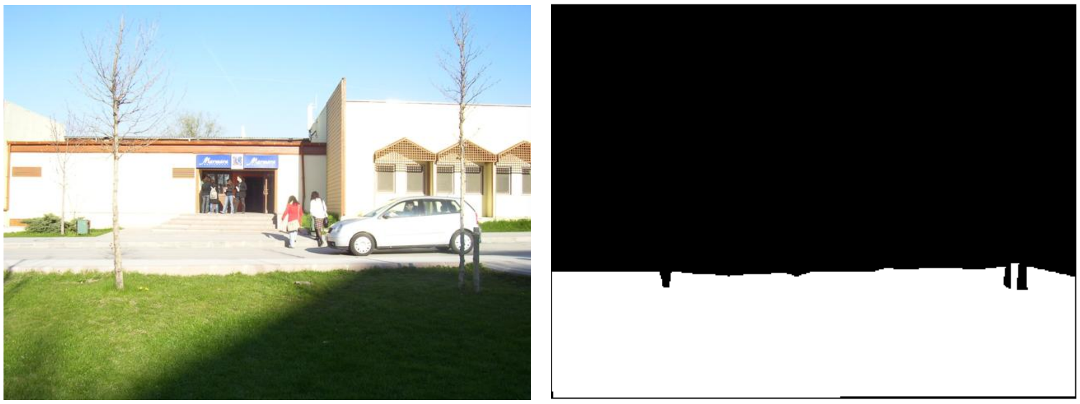
Chapter 2

Object Detectors

The first step for building the contextual scene model is the detection of the individual scene elements. In this chapter, important details regarding object detection are discussed.

Object detection is the localization and classification of objects found in the scene. Since each object class carries different characteristics, there is no perfect solution which works for every class. In other words, a universal object detector does not exist. This fact leads to the idea of designing different type of detectors for different types of objects. For example, some objects such as *cars* have distinctive shape properties while their color and texture features demonstrate diversity. Thus, designing a car detector that depends on color features does not make sense. It is better to make a detector that utilizes the shape information to recognize a car. On the other hand, for the *grass* class, considering shape features is not meaningful. This time, a color based detector may work better because grass tends to appear green. Hence, it is clear that the characteristics of a particular class must be taken into account during detector design.

Although there are differences in the design of detectors, their outputs should be compatible so that they can be handled identically by our framework. This can be achieved by the following requirements. Firstly, the detectors used in our framework must report the object they found as a binary image mask associated



(a) Input image. (b) Detection mask corresponding to grass.

Figure 2.1: Sample detection mask.

with a detection confidence score. A sample detection mask is shown in Figure 2.1. Secondly, the confidence scores must be values between 0 and 1, so they can be interpreted as class membership probabilities.

To represent these probabilities, we use the notation $P(X_i = c|D_i)$. Here, X_i is the variable for the object class label and D_i is the detector type. Therefore, $P(X_i = c|D_i)$ is the probability of assigning class label c to the i'th object that is detected by a detector of type D_i. Here, detector type, D_i, has an important role. According to the value of D_i, the domain of the object class label variable, X_i, is determined. Let D_i be an object detector that can detect objects of classes $c_1, c_2, \ldots, c_k$. In this case, the domain of X_i is defined as the set $\mathcal{C}_i = \{c_1, c_2, \ldots, c_k, unknown\}$. Here, *unknown* is also crucial because it corresponds to the case of not being able to call an object a member of the set of classes detectable by D_i. The *unknown* label will be used to reject a detection if it is not compatible with contextual agreement in the scene in Chapter 4.

These definitions lead to a final requirement of total probability as

$$\sum_{u \in \mathcal{C}_i} P(X_i = u|D_i) = 1. \tag{2.1}$$

This requirement can be used to compute the probability of X_i being *unknown* as

$$P(X_i = unknown|D_i) = 1 - \sum_{u \in \mathcal{C}'_i} P(X_i = u|D_i) \tag{2.2}$$

where $\mathcal{C}'_i = \mathcal{C}_i - \{unknown\}$.

An example regarding the requirements can be given as follows. Assume that the i'th object is detected by a $flower$ detector which can recognize the classes $rose$ and $tulip$. In this case, $D_i = flower$ and $\mathcal{C}_i = \{rose, tulip, unknown\}$. This detector must report the following 3 probabilities beside the detection mask corresponding to the i'th object:

1. $P(X_i = rose|D_i = flower) = p_{rose}$,
2. $P(X_i = tulip|D_i = flower) = p_{tulip}$,
3. $P(X_i = unknown|D_i = flower) = 1 - p_{rose} - p_{tulip}$.

After discussing the detector requirements of our framework, explanations about specific detector designs are given in the rest of this chapter. Although any kind of object detector satisfying the requirements about the mask and the score can be integrated into our framework, we handle three types of detectors in this work. Detectors regarding object classes with discriminative shape properties are described in Section 2.1. Then, in the Section 2.2, pixel classification based approach is presented. Finally, an object detection approach using surface orientations is explained in Section 2.3.

2.1 Shape Based Object Detectors

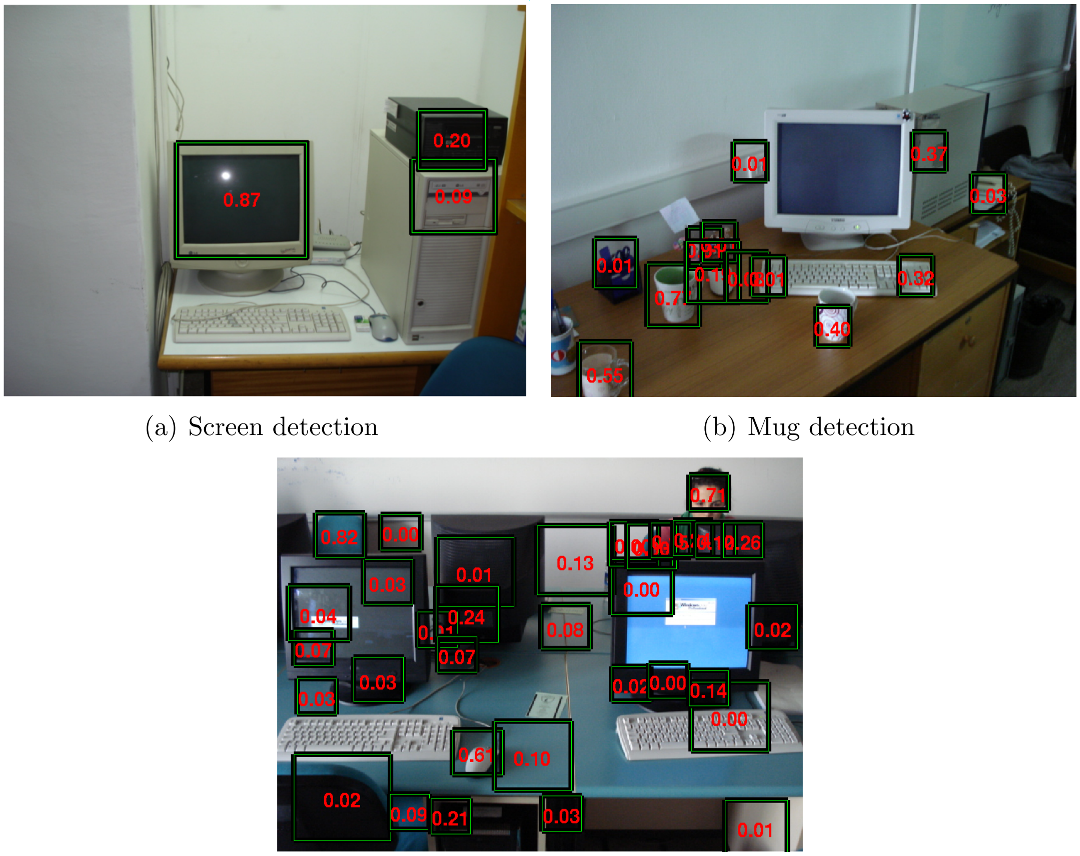
The details of shape based object detectors are explained in this section. These detectors are useful in finding objects whose color and texture features demonstrate changes across different instances while shape remains nearly unchanged. Objects like car, person and window can be put in this group. In order to deal with such objects, we use two different approaches in our framework:

1. Object detection with boosting [32],
2. Object detection based on histograms of oriented gradients (HOG) [7].

The boosting-based detector [32] uses combination of several weak detectors to build a strong detector. Each weak detector performs template matching using normalized cross correlation between the input image and a patch extracted from the training samples. Since patch - object center relationships are available in the training samples, using the correlation output, weak detector votes for the object center. Consequently, weighted combinations of these votes are computed and used to fit bounding boxes on candidate objects. Besides, detection confidence scores (class membership probabilities) are reported based on the votes. Original version of this detector works only in a single scale of the image. In order to increase the applicability of the detector, we implemented a multi-scale version using Gaussian pyramids. Note that this is a single class object detector. Thus, a separate detector for each object class is learned using training examples represented using bounding boxes of sample objects.

Sample boosting-based detections are shown in Figure 2.2. It is clear that in addition to locating a target object successfully, the detector also reports many false alarms. For objects like mouse that usually looks like a small blob, the number of false alarms can actually be very high. We will show that our framework is good at reducing the false alarm rate by eliminating the detections that are contextually inconsistent. Therefore, we prefer to use these simple detectors that usually do not miss a target object at the expense of several false matches.

The other method we use is HOG based detection [7]. In this method, a sliding window approach is used to detect objects. The image tile under the window is classified and assigned a detection confidence score. During classification, first of all, HOG features are extracted by dividing the tile into small cells and computing 1D histogram of edge orientations or gradient directions for each cell. To obtain the HOG descriptor of a tile, histograms obtained from the cells are combined. Finally, this descriptor is used in classification of the current tile under the detection window. If a tile is found to be a target object, then it will be reported as a bounding box associated with a confidence score. Note that for better detection outputs, this detector is also run in multiple scales of the input image. Figure 2.3 shows sample detections performed by HOG based detectors.

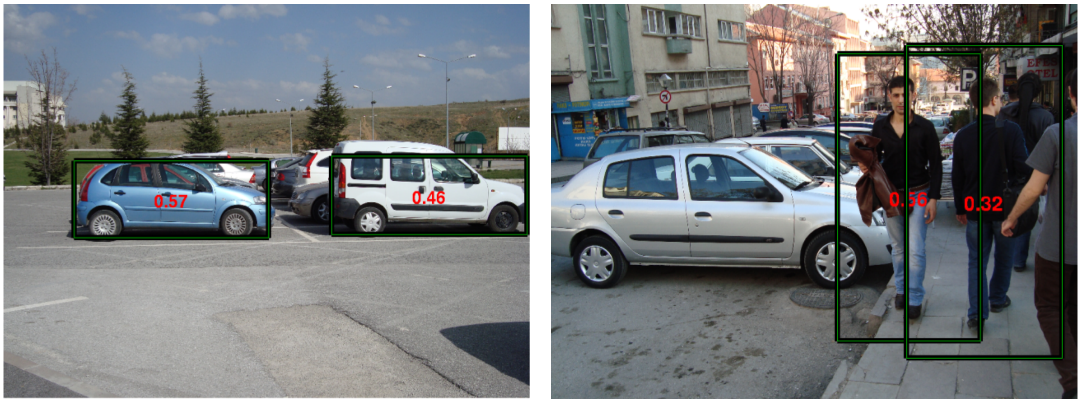


(a) Screen detection

(b) Mug detection

(c) Mouse detection

Figure 2.2: Boosting based detection examples.



(a) Car detection (b) Person detection

Figure 2.3: HOG based detection examples.

In contrast to high false alarm rates of boosting based detectors, HOG based detectors do not tend to report false detections. Although this seems like an advantage, the possibility of missing a target object is higher. Nevertheless, for specific object classes like *car* and *person*, we observed that the output of HOG based detectors are more meaningful than the outputs of the boosting based detectors.

Although there are differences in these two shape based detection algorithms, their outputs are identical: bounding boxes and scores. In order to make the detectors compatible with our framework, the bounding boxes are represented as binary image masks.

Another similarity between two approaches is that both detectors work for the single class case. Thus, for the i'th object, the set $\mathcal{C}_i$ (possible values of X_i) is $\{c, unknown\}$ where c is the particular object class. Then, probabilities reported by the detector should be $P(X_i = c|D_i = c)$ and $P(X_i = unknown|D_i = c)$. Using (2.2), $P(X_i = unknown|D_i = c)$ is calculated as

$$P(X_i = unknown|D_i = c) = 1 - P(X_i = c|D_i = c). \tag{2.3}$$

For example, consider the leftmost car shown in Figure 2.3(a). In this case, $P(X_i = car|D_i = car) = 0.57$, so $P(X_i = unknown|D_i = car)$ must be equal to 0.43.

2.2 Pixel Classification Based Object Detectors

After explaining shape based detection methods, details about pixel classification based approach are given in this section. These detectors are useful in detecting objects whose pixel level features (e.g. color, texture) are discriminative for identification. Example classes having this property are *grass*, *soil*, *sky*, etc. In an image, regions corresponding to such objects do not tend to demonstrate a constant shape. Thus, shape based detectors are not good at finding these objects. Instead, in this case, a bottom-up approach where objects are detected by grouping pixels with similar features is used.

At the heart of this method lies pixel classification. So, first of all, a pixel level feature must be defined. For example, *sky* is made up of blue and white pixels. Thus, choosing RGB as the pixel level feature is intuitive. Secondly, a classifier must be trained based on the chosen feature. This classifier should output a probability of being a pixel of a given object class. In *sky* detection case, this classifier is expected to report high probabilities when RGB values of a blue or white pixel is given. In order to create such classifiers, in our framework, we use one-class classification using the mixture of Gaussians (MoG) distribution as explained in [30]. It utilizes expectation-maximization algorithm to estimate the mixture weights, means and covariances.

The posterior probability of being a target pixel, $P(w_T|y)$, is computed as

$$\begin{aligned} P(w_T|y) &= \frac{p(y|w_T)P(w_T)}{p(y|w_T)P(w_T) + p(y|w_O)P(w_O)} \\ &= \frac{p(y|w_T) \times 0.5}{p(y|w_T) \times 0.5 + p(y|w_O) \times 0.5} \\ &= \frac{p(y|w_T)}{p(y|w_T) + p(y|w_O)} \end{aligned} \tag{2.4}$$

where y is a pixel level feature, w_T represents the target class, w_O is the case of being an outlier (non-target) pixel. Then, $p(y|w_T)$ is the MoG distribution learned using examples and $p(y|w_O)$ is the uniform distribution assumed for the outlier pixels. Finally, $P(w_T)$ and $P(w_O)$ are the prior probabilities for target and outlier pixels, respectively. In our framework, prior probabilities are assumed to be 0.5.

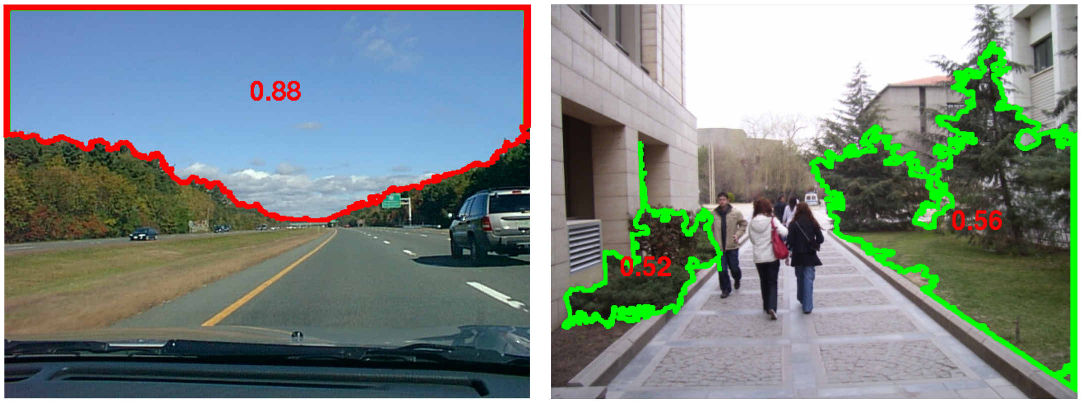
We have explained the training and usage of the pixel level classifiers. Recall that our main goal is object detection. For this purpose, firstly, features corresponding to each pixel of the given image are extracted. Using these features and (2.4), each pixel is assigned a probability of being a target object class pixel. Then, pixels with probabilities under a certain threshold are labeled as background. Afterwards, connected components among the remaining forground pixels are extracted. Connected components (regions) with number of pixels less than a threshold are also labeled as background. Then, holes (background pixels) inside each region are filled. These final regions correspond to the image masks, in other words, target object detections. Recall that our framework also requires detection confidence scores. In order to obtain them, posterior probabilities of each pixel of the detected region are averaged and used as confidence scores (class membership probabilities) as

$$P(X_i = c|D_i = c) = \frac{1}{M_i}\sum_{j=1}^{M_i} P(c|y_j). \tag{2.5}$$

Here, M_i is the number of pixels in the object (region) and $P(c|y_j)$ is the posterior probability of the j'th pixel of the region. Then, $P(X_i = unknown|D_i = c)$ can be computed as

$$P(X_i = unknown|D_i = c) = 1 - P(X_i = c|D_i = c). \tag{2.6}$$

Although this detector seems to be a single class detector, it can easily be extended to detect multiple classes. Consider the classes *grass* and *tree*. When RGB pixel level features of both classes are taken into account, they are not easily distinguishable. In this case, it is better to create a *vegetation* detector. Let vegetation detector work as a single class pixel classification based object detector. Then, we have the following outputs: $P(X_i = vegetation|D_i = vegetation)$, $P(X_i = unknown|D_i = vegetation)$, and the image masks corresponding to *vegetation* objects. Given these probabilities, $P(X_i = grass|D_i = vegetation)$



(a) Sky detection.

(b) Vegetation detection.

Figure 2.4: Pixel classification based object detection examples.

can be computed as

$$\begin{aligned} P(X_i = grass|D_i) &= P(X_i = vegetation, X_i = grass|D_i) \\ &= P(X_i = grass|X_i = vegetation, D_i)P(X_i = vegetation|D_i) \\ &= P(X_i = grass|X_i = vegetation)P(X_i = vegetation|D_i). \end{aligned} \tag{2.7}$$

Note that every grass is also a vegetation object. Thus, event of $\{X_i = grass\}$ is identical to the event of $\{X_i = vegetation \ \cap X_i = grass\}$.

Similarly, $P(X_i = tree|D_i = vegetation)$ is calculated as

$$P(X_i = tree|D_i) = P(X_i = tree|X_i = vegetation)P(X_i = vegetation|D_i). \tag{2.8}$$

Probabilities, $P(X_i = grass|X_i = vegetation)$ and $P(X_i = tree|X_i = vegetation)$, are estimated using a training set with manual object labels as

$$P(X_i = grass|X_i = vegetation) = \frac{\#(vegetation \text{ objects labeled as } grass)}{\#(vegetation \text{ objects})} \tag{2.9}$$

and

$$P(X_i = tree|X_i = vegetation) = \frac{\#(vegetation \text{ objects labeled as } tree)}{\#(vegetation \text{ objects})}, \tag{2.10}$$

respectively.

In multiclass version, computation of $P(X_i = unknown|D_i = c)$ is still performed using (2.6).

Sample pixel based object detection outputs are shown in Figure 2.4.

2.3 Surface Orientation Based Object Detectors

We have discussed the cases where objects have discriminative features like specific shape, color and texture. However, there are also object classes which do not demonstrate a pattern in terms of these features. Instances of such classes usually appear in arbitrary but uniform color and texture. For example, wall, table top, floor and ceiling share this property. Consider a circular wooden table and a rectangular plastic table. They seem completely different, so classifying them into same class using specific values of their color, texture or shape features does not make sense. Objects demonstrating this property are called *surface objects* in our framework.

Hoiem [16] developed a method for recovering the surface layout from a single still image. The method labels the regions of an image with geometric classes like *support* and *vertical*. Support regions refer to the objects that are approximately parallel to the ground (eg. road, table tops). Vertical image areas correspond to the objects such as walls, trees, pedestrians and buildings. We use this method to obtain confidence maps for the geometric classes *horizontal* (0 degree) and *vertical* (90 degrees) relative to the ground plane.

Detection of surface objects begins with finding associated image regions. In order to do so, we group the individual pixels into few partitions by applying k-means clustering to the verticality and horizontality confidences of pixels. For k-means clustering, each pixel's horizontality and verticality confidence are concatenated and used as feature vector. After clustering, connected components are extracted using pixels assigned to same group. Components with number of pixels less than a threshold are labeled as background. Remaining ones are kept as image masks showing detected surface objects.

Second step in the detection is the determination of class membership probabilities of the detected objects. For this purpose, again we use the verticality and horizontality confidence maps. First of all, the horizontality and verticality probabilities for each detected surface object are computed by averaging the values in the corresponding regions of the confidence maps. We denote the horizontality and verticality probabilities as $P(G_i = \mathit{horizontal}|D_i = \mathit{surface})$ and $P(G_i = \mathit{vertical}|D_i = \mathit{surface})$, respectively, where G_i represents the geometric class label and D_i represents the type of the detector. The probability of a surface not being horizontal or vertical is computed as

$$\begin{aligned} P(G_i = \mathit{unknown}|D_i = \mathit{surface}) = 1 &- P(G_i = \mathit{horizontal}|D_i = \mathit{surface}) \\ &- P(G_i = \mathit{vertical}|D_i = \mathit{surface}). \end{aligned} \tag{2.11}$$

The horizontal and vertical surfaces can be further divided into categories such as road, table, etc. for horizontal, and wall, building, etc. for vertical. The i'th object's probability of being a surface object of type c can be computed as

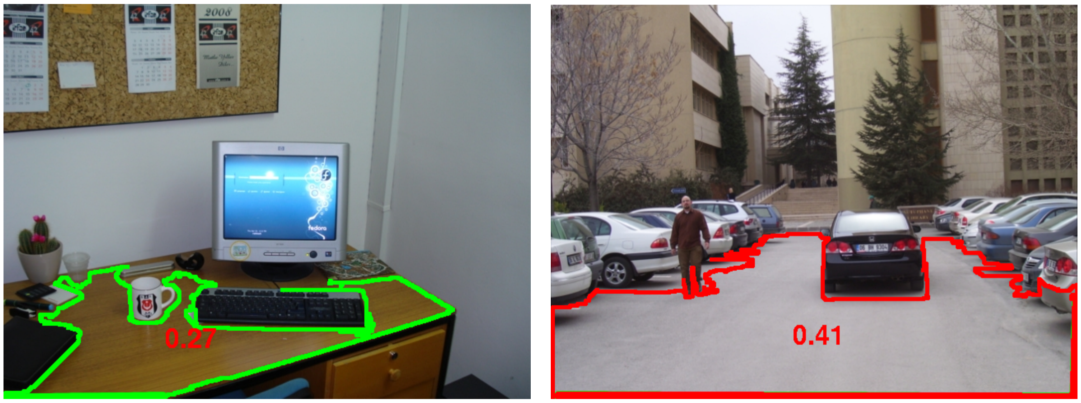
$$\begin{aligned} P(X_i = c|D_i = \mathit{surface}) &= \sum_{G_i} P(X_i = c, G_i|D_i = \mathit{surface}) \\ &= \sum_{G_i} P(X_i = c|G_i)P(G_i|D_i = \mathit{surface}). \end{aligned} \tag{2.12}$$

Note that $P(X_i = c|G_i = \mathit{horizontal}) = 0$ if c is not a horizontal surface type, $P(X_i = c|G_i = \mathit{vertical}) = 0$ if c is not a vertical surface type, and $P(X_i = \mathit{unknown}|G_i = \mathit{unknown})$ is always equal to 1. Then probabilities for horizontal objects are computed as

$$\begin{aligned} &P(X_i = c_h|D_i = \mathit{surface}) \\ &\qquad = P(X_i = c_h|G_i = \mathit{horizontal}) \\ &\qquad\qquad P(G_i = \mathit{horizontal}|D_i = \mathit{surface}) \end{aligned} \tag{2.13}$$

and the probabilities for vertical objects are computed as

$$\begin{aligned} &P(X_i = c_v|D_i = \mathit{surface}) \\ &\qquad = P(X_i = c_v|G_i = \mathit{vertical}) \\ &\qquad\qquad P(G_i = \mathit{vertical}|D_i = \mathit{surface}) \end{aligned} \tag{2.14}$$



(a) Desk detection. (b) Road detection.

Figure 2.5: Surface orientation based detection examples.

where $P(X_i = c_h|G_i = horizontal)$ and $P(X_i = c_v|G_i = vertical)$ are estimated from the percentage of the number of horizontal surface objects labeled as c_h among all horizontal surface objects, and the percentage of the number of vertical surface objects labeled as c_v among all vertical surface objects, respectively. Finally, the possibility of not being able to label a surface object is modeled by the probability

$$\begin{aligned} &P(X_i = unknown|D_i = surface) \\ &\quad = \sum_{G_i} P(X_i = unknown|G_i) P(G_i|D_i = surface) \\ &\quad = P(X_i = unknown|G_i = unknown) P(G_i = unknown|D_i = surface) \\ &\quad = P(G_i = unknown|D_i = surface). \end{aligned} \tag{2.15}$$

Figure 2.5 shows sample surface object detection results.

Chapter 3

Interactions Between Scene Components

Contextual interactions observed in a scene must be incorporated into the decision process about that scene to achieve a better detection performance. A candidate interaction is the co-occurrence of the objects. Learning this contextual information is relatively easy when the data set contains an adequate number of groundtruth class label assignments. However, co-occurrences alone may not be sufficient to encode the whole context in the scene. For example, cars and buildings tend to co-occur, but it is not usual to observe a building located under a car. Suppose these objects are accidentally detected in the scene. Since they are consistent according to the co-occurrence likelihood, the system will favor such unreasonable configuration.

Therefore, a more sophisticated system should consider the relative positions/locations, scales and orientations observed among the objects. This corresponds to the use of spatial realtionships. If a system is aware of the most likely spatial relationships between the objects, it can handle unlikely situations with a higher precision. Algorithms that can model topological (set relationships, adjacency), distance-based (near, far) and relative position-based (above, below, right, left) relationships have been successfully applied to satellite image analysis for improving the classification accuracy [1]. However, relationships that are defined in the 2D image space are not always applicable to generic scene analysis because object relationships in the 3D world can appear in many different configurations in a 2D image. Example images for these configurations are shown in Figure 1.2. In these images, the car with the red marker is parked behind the car with the green marker. However, when the 2D spatial relationships of the cars are interpreted, they seem to have different relative positions. Hence, naming the spatial relationships by just analyzing the relative positions is problematic.

Although both ways of handling context have some disadvantages, they can be combined into a more powerful approach. In our proposed method, co-occurence likelihood's robustness to the wrong estimations of the spatial relationship types and the strength of relative position/location, scale and orientation are handled in the same probabilistic model. For this purpose, three different spatial relationship features are introduced and explained in Section 3.1. Then, we mention how we use these features to estimate object interaction likelihoods in Section 3.2.1. Since we have chosen co-occurrence likelihoods as a baseline approach to be compared with our proposed model during experiments (Chapter 5), we also explain how co-occurrence probabilities can be utilized as interaction likelihoods in Section 3.2.2.

3.1 Spatial Relationship Features

A single spatial relationship feature is not sufficient to describe the observed relationship between objects. Therefore, we use multiple features to encode the interactions. By this way, we can analyze the pairwise interactions from different perspectives and take advantage of different measures.

Feature values are calculated using the binary image masks corresponding to the objects. Since features used in our framework are *relative*, during feature extraction for an object pair, one of them is chosen as the *reference* object.

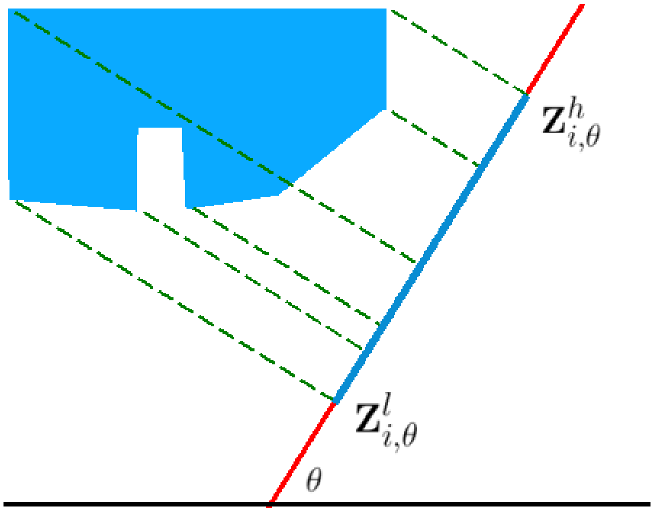


Figure 3.1: Projection of an object onto the line with orientation θ. **Blue region:** object to be projected, **red line:** orientation of projection, **blue line segment:** projected object.

Throughout this section, suppose that spatial relationship features are extracted for object o_i with respect to reference object o_{ref}.

Let $\mathcal{P}_i$ be the set of pixels constituting the object o_i and $\mathbf{Z}_i$ be the $2 \times M_i$ matrix whose columns are the (x, y)-coordinates of o_i's pixels (foreground pixels in the binary image mask) where $M_i = |\mathcal{P}_i|$. Then,

$$\mathbf{Z}_i = \begin{bmatrix} x_{i,1} & x_{i,2} & x_{i,3} & \dots & x_{i,M_i} \\ y_{i,1} & y_{i,2} & y_{i,3} & \dots & y_{i,M_i} \end{bmatrix}. \tag{3.1}$$

Some of the features we will introduce require projections of the objects. Let $\mathbf{A}_\theta$ be the transformation matrix which projects the objects onto the line with orientation θ relative to the horizontal axis as

$$\mathbf{A}_\theta = \begin{bmatrix} \cos\theta & \sin\theta \end{bmatrix}. \tag{3.2}$$

Then, $\mathbf{Z}_{i,\theta}$ is the $1 \times M_i$ vector containing pixel coordinates in the projected space as

$$\mathbf{Z}_{i,\theta} = \mathbf{A}_\theta \mathbf{Z}_i. \tag{3.3}$$

An example projection is given in Figure 3.1. When an object is projected on a line, the image region corresponding to that object is converted to a line segment. For some features, end points of this line segment are required. Lower and higher end points are denoted by $\mathbf{Z}^l_{i,\theta}$ and $\mathbf{Z}^h_{i,\theta}$, respectively as

$$\mathbf{Z}^l_{i,\theta} = \min\{\mathbf{Z}_{i,\theta}\} \tag{3.4}$$

$$\mathbf{Z}^h_{i,\theta} = \max\{\mathbf{Z}_{i,\theta}\}. \tag{3.5}$$

Details of our spatial relationship features are explained in the following subsections.

3.1.1 Oriented Overlaps Feature

Overlap ratios of two regions are widely used features. An example of its usage can be seen in [13]. It is simply calculated by dividing the number of pixels in the overlapping area by the number of pixels in one of the objects. However, this simple feature does not capture the relative location of the participating regions. Thus, it is not sufficient to encode the observed relationship. To overcome this issue, overlap features are mostly used with relative centroid positions. Since centroid of an object with arbitrary shape is not always representative enough to describe the location of the entire object, this combined feature does not work as expected. In addition, the units of overlap ratio and centroid differences are not compatible (while overlap ratio is unitless, centroid feature is measured in pixels). Therefore, the feature space generated by the combination of these features is not meaningful.

Here, we introduce an overlap based feature called *oriented overlaps*. It encodes the relative overlap and position in a unified representation. This feature is extracted as follows. First of all, some orientations of interest are selected. Let the set of these orientations be $\Theta = \{\theta_1, \theta_2, \ldots, \theta_{|\Theta|}\}$. Then, for each element of Θ, $\mathbf{Z}^l_{ref,\theta_j}$ and $\mathbf{Z}^h_{ref,\theta_j}$ are computed, where $1 < j < |\Theta|$. Recall that these represent the lower and higher end points of the line segment corresponding to the object o_{ref} in the projection space, respectively. Let the set $\mathcal{P}_{i,ref,\theta_j}$ be the

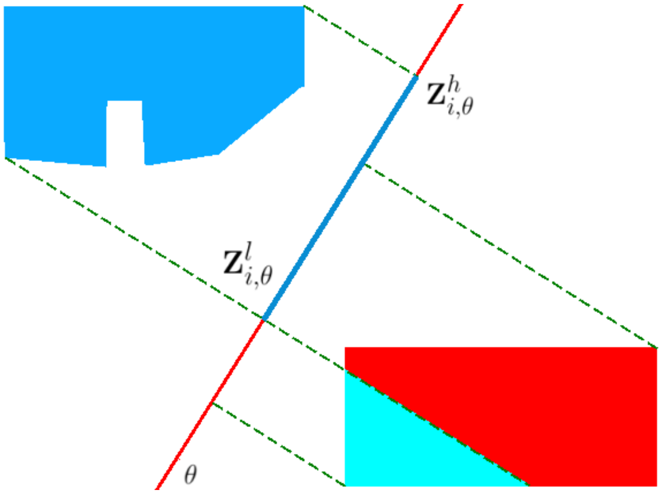


Figure 3.2: The computation of ρ_{i,ref,θ_j}. Here, the reference object, o_{ref}, is shown as a blue region. The object whose oriented overlaps feature is being computed, o_i, is shown as a red-cyan region. The red portion of o_i is composed of the pixels whose projections are values in the $[\mathbf{Z}^l_{ref,\theta_j}, \mathbf{Z}^h_{ref,\theta_j}]$ interval (the blue line segment). Thus, ρ_{i,ref,θ_j} is calculated as (number of o_i's red pixels)/(total number of o_i's pixels).

subset of o_i's pixels whose projections are values in the $[\mathbf{Z}^l_{ref,\theta_j}, \mathbf{Z}^h_{ref,\theta_j}]$ interval. Then, the overlap ratio, ρ_{i,ref,θ_j}, observed in orientation θ_j is calculated as

$$\rho_{i,ref,\theta_j} = \frac{|\mathcal{P}_{i,ref,\theta_j}|}{M_i}. \tag{3.6}$$

Illustration corresponding to the computation of ρ_{i,ref,θ_j} is shown in Figure 3.2.

The oriented overlaps feature vector, $\mathbf{O}_{i,ref}$, is formed by concatenating the features for all orientations as

$$\mathbf{O}_{i,ref} = \begin{bmatrix} \rho_{i,ref,\theta_1} & \rho_{i,ref,\theta_2} & \rho_{i,ref,\theta_3} & \dots & \rho_{i,ref,\theta_{|\Theta|}} \end{bmatrix}. \tag{3.7}$$

In our framework, to keep the feature space simple enough, we chose Θ as the set $\{0, 90\}$. Although information coming from other orientations is also important, in order to avoid high dimensionality problems, we preferred 0 and 90 degree orientations.

In Figure 3.3, feature space of oriented overlaps feature is shown. Blue dots

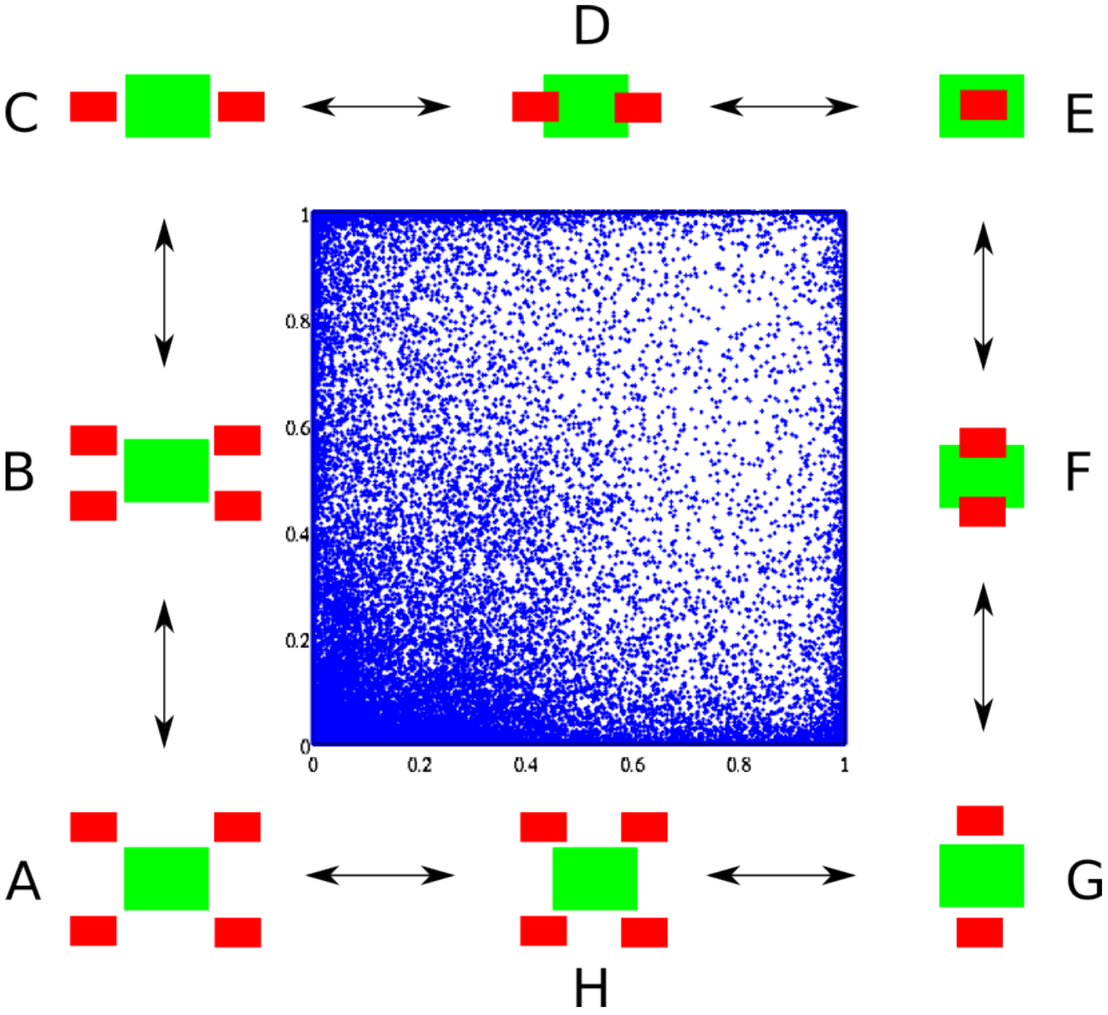


Figure 3.3: Oriented overlaps feature space. While x-axis shows values of $\rho_{i,ref,0}$, y-axis corresponds to $\rho_{i,ref,90}$. Features are extracted for the red object (o_i) with respect to the green object (o_{ref}). See text for more details.

are the features extracted using object pairs found in one of the data sets (a subset of LabelMe [27]) we use in the experiments. The double-headed arrows show how the relationship between the red (o_i) and green (o_{ref}) boxes change as the feature values vary in the specified direction. Note that the range of feature values is $[0, 1]$ for both $\rho_{i,ref,0}$ and $\rho_{i,ref,90}$.

Consider the boxes shown in group A. Using any of the four red boxes in feature extraction with the green box (the reference object) results in the same feature vector, $(0, 0)$. As $\rho_{i,ref,0}$ (x-axis) increases, the positions of the red boxes change as shown in the transitions A → H, H → G, C → D and D → E. Similarly, as $\rho_{i,ref,90}$ (y-axis) value increases, the positions change as shown in the transitions A → B, B → C, G → F and F → E.

Note that although there are usually more than one possible relative position for a single oriented overlaps feature vector, the relationship observed in each position is similar regardless of the direction.

3.1.2 Oriented End Points Feature

Although the oriented overlaps feature is informative about the relative positions, it cannot distinguish the spatial relationships in which direction is important. For example, a mouse can be found either on the left or right of a keyboard. It is clear that direction of the relationship is not significant in the keyboard and mouse case, so the oriented overlaps feature is sufficient to encode the relationship between them. However, the situation is different for the grass and sky classes. In a natural scene, the sky is expected to appear above the grass. But, if the oriented overlaps feature is used, observing sky below the grass will be encoded similarly. The reason is that the oriented overlaps feature encodes only the amount of overlap (a scalar value, i.e. no direction information) in different orientations.

In order to distinguish the relationships in which direction is also important besides the orientation, we define another feature called *oriented end points*. Extraction of this feature is as follows. First of all, an orientation of interest, θ,

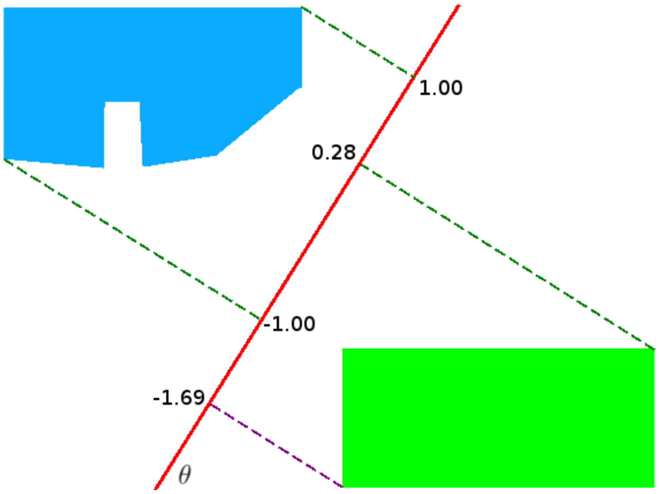


Figure 3.4: An example computation of $\mathbf{E}_{i,ref,\theta}$. Here, the reference object, o_{ref}, is shown as a blue region. The object whose oriented end points feature is being computed, o_i, is shown as a green region. After the projection and normalization steps, the end points of the objects take the values shown in the figure. Thus, $\mathbf{E}_{i,ref,\theta}$ is computed as $(-1.69, 0.28)$.

is selected. Then values, $\mathbf{Z}^l_{i,\theta}$, $\mathbf{Z}^h_{i,\theta}$, $\mathbf{Z}^l_{ref,\theta}$ and $\mathbf{Z}^h_{ref,\theta}$, are calculated. Recall that these values are the end points of the line segments representing the objects in the space of projection.

This feature is expected to encode the relative positions of the line segments. For this purpose, a transformation function, Γ, that normalizes the reference object's line segment is defined as

$$\Gamma(z) = \frac{2z - \mathbf{Z}^l_{ref,\theta} - \mathbf{Z}^h_{ref,\theta}}{\mathbf{Z}^h_{ref,\theta} - \mathbf{Z}^l_{ref,\theta}} \tag{3.8}$$

where $\Gamma(\mathbf{Z}^l_{ref,\theta}) = -1.00$ and $\Gamma(\mathbf{Z}^h_{ref,\theta}) = 1.00$. Using the same transformation function, oriented end points feature, $\mathbf{E}_{i,ref,\theta}$, is constructed as

$$\mathbf{E}_{i,ref,\theta} = \left[\Gamma(\mathbf{Z}^l_{i,\theta}) \quad \Gamma(\mathbf{Z}^h_{i,\theta}) \right] . \tag{3.9}$$

Since this is a normalized feature, it encodes both the relative position and the relative scale in the same representation.

An example computation of $\mathbf{E}_{i,ref,\theta}$ is shown in Figure 3.4.

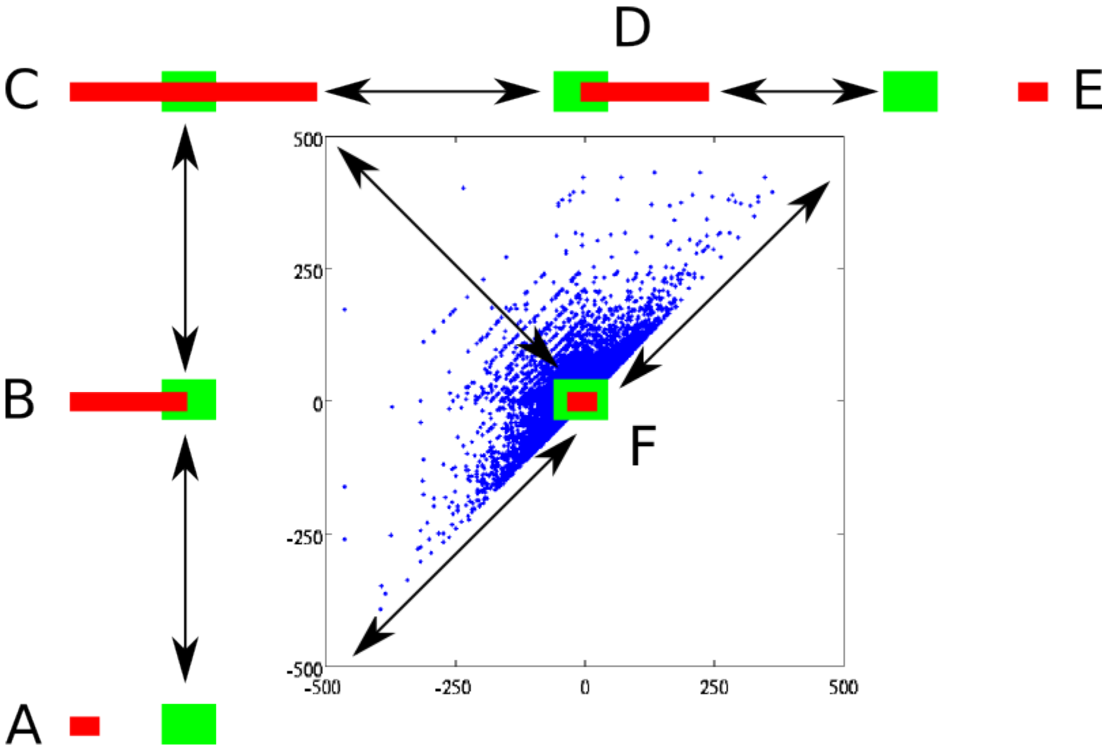


Figure 3.5: Oriented end points feature space for $\theta = 0$. While x-axis shows values of $\Gamma(\mathbf{Z}^{l}_{i,0})$, y-axis corresponds to $\Gamma(\mathbf{Z}^{h}_{i,0})$. Features are extracted for the red object (o_i) with respect to the green object (o_{ref}). See text for more details.

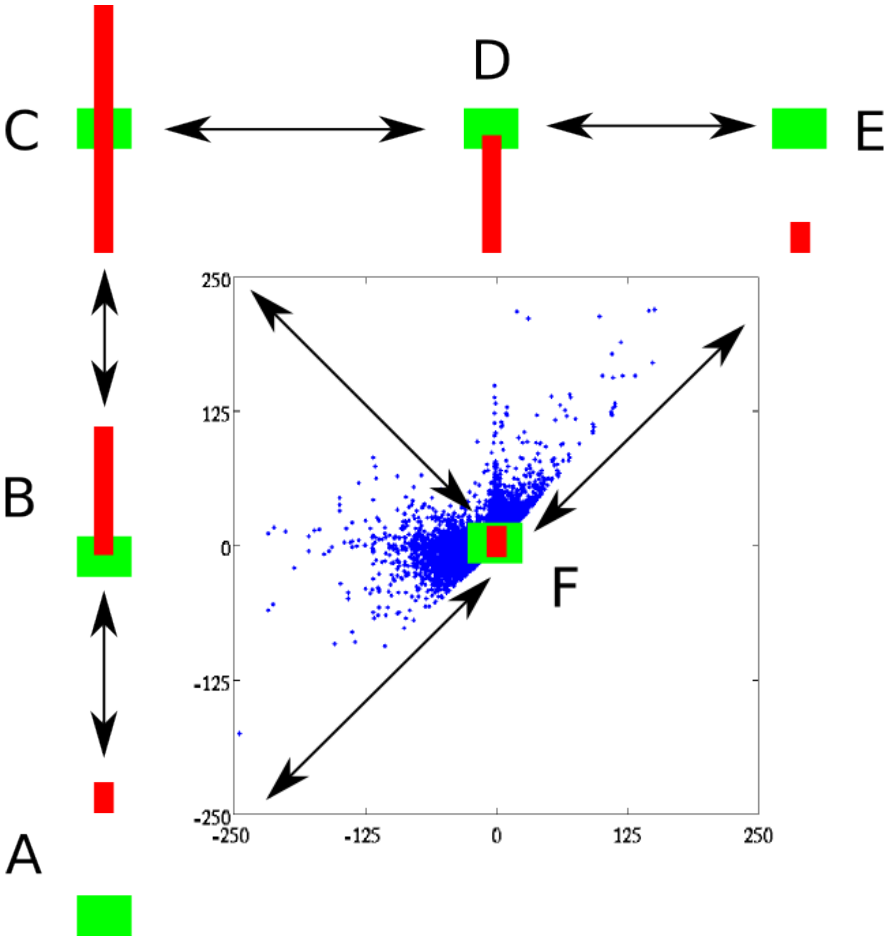


Figure 3.6: Oriented end points feature space for $\theta = 90$. While x-axis shows values of $\Gamma(\mathbf{Z}^l_{i,90})$, y-axis corresponds to $\Gamma(\mathbf{Z}^h_{i,90})$. Features are extracted for the red object (o_i) with respect to the green object (o_{ref}). See text for more details.

In our framework, we extract oriented end points features for orientations 0 and 90. Therefore, there are two feature vectors, $\mathbf{E}_{i,ref,0}$ and $\mathbf{E}_{i,ref,90}$, to be extracted for each object pair.

Figure 3.5 shows the feature space for oriented end points feature where $\theta = 0$. Blue dots are the features extracted using object pairs found in the subset of LabelMe images. The double-headed arrows show how the relationship between the red (o_i) and green (o_{ref}) boxes change as the feature values vary in the specified direction. Note that $\Gamma(\mathbf{Z}^l_{i,0}) \leq \Gamma(\mathbf{Z}^h_{i,0})$ and the range of feature values is $(-\infty, +\infty)$ for both $\Gamma(\mathbf{Z}^l_{i,0})$ and $\Gamma(\mathbf{Z}^h_{i,0})$.

As the $\Gamma(\mathbf{Z}^l_{i,0})$ (x-axis) increases, the size of the red object decreases as shown in transitions C $\rightarrow$ D and D $\rightarrow$ E. On the other hand, when $\Gamma(\mathbf{Z}^h_{i,0})$ (y-axis) increases, the size of the object increases as shown in transitions A $\rightarrow$ B and B $\rightarrow$ C. Note that besides changes in size, the position of the red object also varies in those transitions. The diagonal transitions like A $\rightarrow$ F and F $\rightarrow$ E affect only the position of the red object. On the contrary, the transition F $\rightarrow$ C changes the size instead of the position.

The feature space of oriented end points where $\theta = 90$ is available in Figure 3.6. This figure can be similarly interpreted as the feature space shown in Figure 3.5.

3.1.3 Horizontality Feature

We have explained two spatial relationship features that try to encode relative position and scale observed between objects. Both features are extracted using the observations from the 2D image space. However, objects are found in a 3D environment in real world. Therefore, in order to encode a relationship, information coming from 3D space is also useful. The *horizontality* feature is defined for this purpose. This feature measures the relative horizontality confidence of o_i with respect to o_{ref}. Recall the Hoiem's technique [16] to find surface layout from a single still image. We use this technique to find surface objects as explained in

Section 2.3. In order to extract the horizontality feature, the horizontality confidence for o_i and o_{ref} are computed by averaging the values in the corresponding regions of the horizontality confidence map. Let H_i and H_{ref} be the confidence values of the objects. The scalar horizontality feature, $\mathbf{H}_{i,ref}$, is simply calculated as

$$\mathbf{H}_{i,ref} = H_i - H_{ref}. \tag{3.10}$$

There is no need to compute a verticality feature, because there is a high correlation between horizontality and verticality. In [16], for each image, confidence maps for three main classes are extracted: support, vertical, and sky, where the sum of three confidence values for a pixel is equal to 1. Therefore, relative verticality is equal to

$$\begin{aligned} V_i - V_{ref} &= (1 - H_i - Sky_i) - (1 - H_{ref} - Sky_{ref}) \\ &= -(H_i - H_{ref}) - (Sky_i - Sky_{ref}) \\ &= -\mathbf{H}_{i,ref} - (Sky_i - Sky_{ref}). \end{aligned} \tag{3.11}$$

For objects other than sky, $(Sky_i - Sky_{ref}) \simeq 0$, so relative verticality is approximately equal to $-\mathbf{H}_{i,ref}$. Therefore, using horizontality feature is sufficient.

Feature space for $\mathbf{H}_{i,ref}$ is simple. It takes values from -1 to 1 inclusively. As the value approaches to -1, verticality of o_i increases. Conversely, values close to 1 shows that object is more horizontal. For example, $\mathbf{H}_{screen,desk}$ is expected to have a value closer to -1.

3.2 Object Interaction Likelihood

The meaningfulness of the interactions (co-occurrence, relative position, location and pose) between objects can be measured in terms of *interaction likelihoods* that are computed with respect to the training examples. It is denoted as $p(S_{i,j}|X_i, X_j)$. This is the probability of observing $S_{i,j}$, the interaction between two objects, o_i and o_j, having object class labels X_i and X_j, respectively.

We focus on how the spatial relationship features and the co-occurrence probabilities are utilized to represent the object interactions in Sections 3.2.1 and 3.2.2, respectively.

3.2.1 Spatial Relationship Feature Based Interaction Likelihood

In our proposed method, we use the spatial relationship features to estimate how likely a particular interaction between two objects is by learning probabilistic models for each object class pair using these features. Then, for a new object pair, likelihood of the interaction is computed using the associated model. The value of $p(S_{i,j}|X_i, X_j)$ is calculated as

$$p(S_{i,j}|X_i, X_j) = \prod_{\omega_{i,j} \in \Omega_{i,j}} p(\omega_{i,j}|X_i, X_j) \tag{3.12}$$

where $\Omega_{i,j}$ is the set of spatial relationship features extracted for the (o_i, o_j) pair. In the full probabilistic model, $\Omega_{i,j}$ is taken as the set $\{\mathbf{O}_{i,j}, \mathbf{E}_{i,j,0}, \mathbf{E}_{i,j,90}, \mathbf{H}_{i,j}\}$. Then, $p(S_{i,j}|X_i, X_j)$ is computed as

$$p(S_{i,j}|X_i, X_j) = p(\mathbf{O}_{i,j}|X_i, X_j)p(\mathbf{E}_{i,j,0}|X_i, X_j)p(\mathbf{E}_{i,j,90}|X_i, X_j)p(\mathbf{H}_{i,j}|X_i, X_j). \tag{3.13}$$

Note that using full model is not mandatory. Any subset of the full set can be used to estimate interaction likelihoods.

Individual $p(\omega_{i,j}|X_i, X_j)$'s are calculated as smoothed histogram estimates. For this purpose, from a training set where individual objects are manually labeled, we collect feature vectors for each object class pair like (*screen*, *desk*), (*sky*, *building*), (*keyboard*, *keyboard*), (*grass*, *road*), etc. Since our feature vectors are extracted with respect to a reference object, we also collect features for (*desk*, *screen*), (*building*, *sky*) and (*road*, *grass*) pairs. This means that we extract two feature vectors for each object class pair due to possible asymmetry in the corresponding relations. Then, we estimate the object pair conditional density of

Table 3.1: Properties of the spatial relationship feature histograms.

Feature	**#Bins**	**Range**	**Gaussian Kernel**
Oriented overlaps	20×20	$\rho_{i,ref,0} \in [0, 1]$ $\rho_{i,ref,90} \in [0, 1]$	3×3 $\sigma = 0.75$
Oriented end points $(\theta = 0)$	20×20	$\Gamma(\mathbf{Z}^l_{i,0}) \in [-50, 50]$ $\Gamma(\mathbf{Z}^h_{i,0}) \in [-50, 50]$	3×3 $\sigma = 0.75$
Oriented end points $(\theta = 90)$	20×20	$\Gamma(\mathbf{Z}^l_{i,90}) \in [-50, 50]$ $\Gamma(\mathbf{Z}^h_{i,90}) \in [-50, 50]$	3×3 $\sigma = 0.75$
Horizontality	1×20	$\mathbf{H}_{i,ref} \in [-1, 1]$	1×3 $\sigma = 0.75$

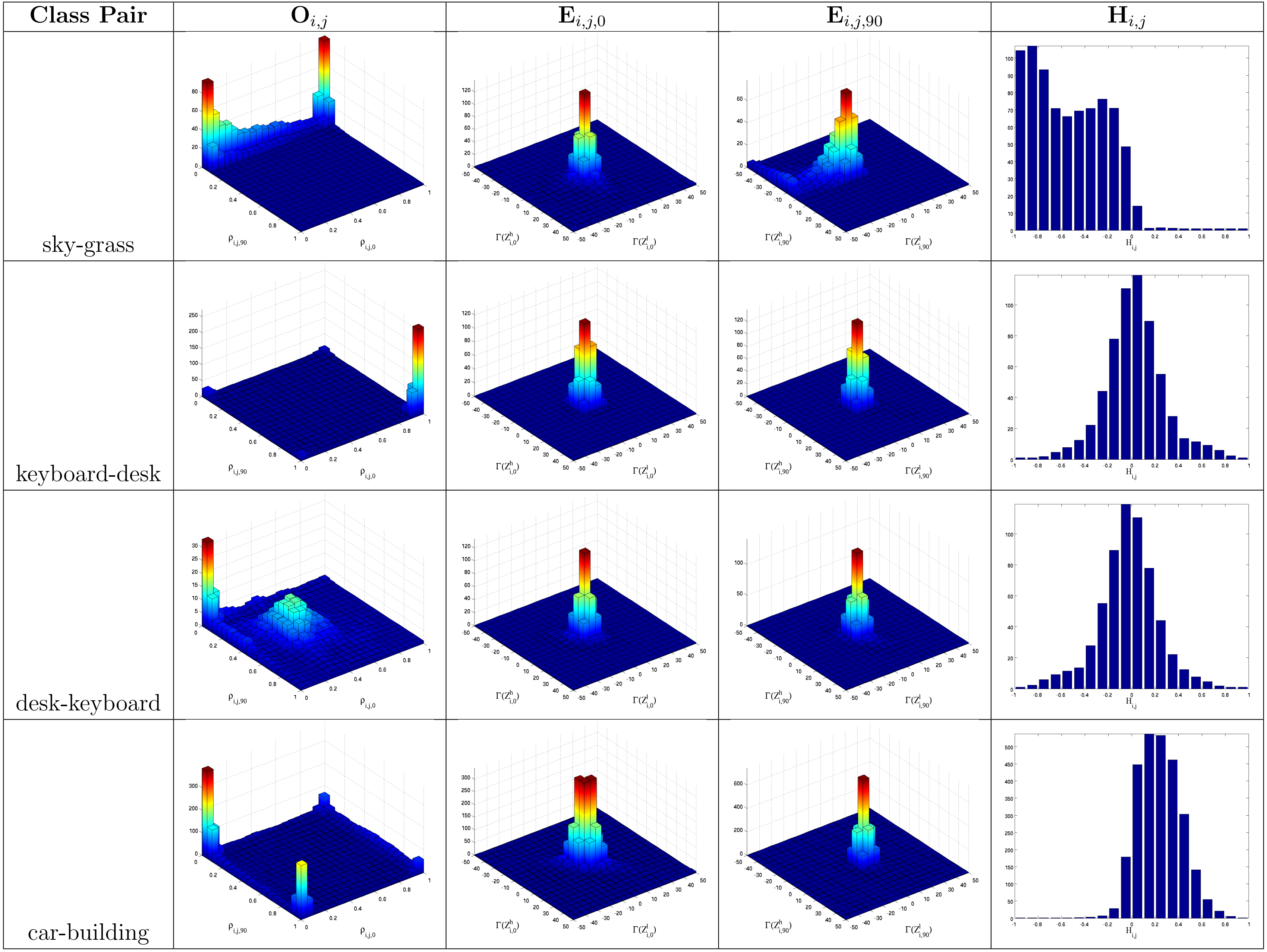
these features, $p^{\omega}_{u,v}(\omega_{i,j})$, using the non-parametric histogram estimate as

$$p^{\omega}_{u,v}(\omega_{i,j}) = \frac{k^{\omega}_{u,v}(\omega_{i,j})}{n^{\omega}_{u,v} V^{\omega}_{u,v}} \tag{3.14}$$

where $k^{\omega}_{u,v}(\omega_{i,j})$ is the value in the histogram bin where $\omega_{i,j}$ falls in, $n^{\omega}_{u,v}$ is the sum of the values in all bins and $V^{\omega}_{u,v}$ is the bin volume. Note that the histogram estimates of (*building*, *sky*) and (*sky*, *building*) pairs are not the same.

The original histograms may not be appropriate for the estimation of $p^{\omega}_{u,v}(\omega_{i,j})$ values due to zero frequency problem and sharp changes in the adjacent bins. Thus, we assume a uniform prior by incrementing each histogram bin by one in order to avoid zero frequency problem observed due to zero counts in the histograms. Moreover, we avoid sharp changes in the adjacent bins by smoothing the histogram using a Gaussian kernel. Consequently, we obtain the final smoothed histograms that are used to calculate the interaction likelihood for a new object pair. Table 3.1 summarizes the properties of the histograms for oriented overlaps, oriented end points and horizontality features. Note that although the range of oriented end points feature values is $(-\infty, +\infty)$, we limit the range of these values to $[-50, 50]$ in the histograms. The reason is that feature vectors out of this range are not observed very frequently and the number of bins does not need to be significantly increased for such rare cases. Thus, any value less than -50 is mapped to -50 and any value greater than 50 is replaced by 50.

Figure 3.7: Example smoothed histograms extracted from the LabelMe data set.



Example smoothed histograms extracted from the LabelMe data set are shown in Figure 3.7. Since a sky object is usually observed above a grass object, their overlap ratio in orientation 90 is very small. Thus, the oriented overlaps feature histogram for the sky-grass pair contains high values only in the bins that correspond to low overlap ratios in 90 degree orientation. Similarly, a keyboard is mostly found on a desk, i.e. the oriented overlaps feature measured using a keyboard with respect to a desk is usually close to $(1, 1)$. Hence, the corresponding histogram has its highest values in the bins close to $(1, 1)$ as expected. However, this tendency is not observed in the oriented overlaps feature histogram for the desk-keyboard pair. In this case, the features are more diverse due to the high variability of the overlap ratios.

Unlike the oriented overlaps feature histograms, ones based on the oriented end point features resemble each other. In these histograms, features are mostly accumulated in the center of the feature space. However, for the pairs like sky-grass, the end point variability can cause an increase in the values of the bins that are far from the center.

Interpretation of the horizontality feature histograms is more intuitive when compared to the histograms of the other features. For example, a grass object is expected to be more horizontal than a sky object. Thus, the sky-grass horizontality feature histogram has its highest values for the bins corresponding to negative relative horizontality. On the other hand, since a keyboard and a desk are both horizontally oriented objects, their histogram reaches its maximum in the bins around 0 relative horizontality.

The smoothed histograms are used to calculate the final interaction likelihood based on a particular feature, $\omega_{i,j}$, as

$$p(\omega_{i,j}|X_i = u, X_j = v) = \min\{p^{\omega}_{u,v}(\omega_{i,j}), p^{\omega}_{v,u}(\omega_{j,i})\} \tag{3.15}$$

where $p^{\omega}_{u,v}(\omega_{i,j})$ is the smoothed histogram estimate using the histogram for the (u, v) class pair, while $p^{\omega}_{v,u}(\omega_{j,i})$ is the smoothed histogram estimate using the histogram for the (v, u) pair. For example, the interaction likelihood according to the oriented overlaps feature is calculated as

$$p(\mathbf{O}_{i,j}|X_i = screen, X_j = desk) = \min\{p^{\mathbf{O}}_{screen,desk}(\mathbf{O}_{i,j}), p^{\mathbf{O}}_{desk,screen}(\mathbf{O}_{j,i})\}. \tag{3.16}$$

Note that in (3.15), we take the minimum of two asymmetric smoothed histogram estimates. By this way, being in a likely interaction can only be achieved by having higher minimum density estimate.

Recall that context has effects at different levels: semantic, spatial configuration and pose [23]. The semantic context corresponds to the co-occurrence tendencies of objects. Our interaction likelihoods give low values for objects that do not tend to be found in the same image. Thus, our method encodes the semantic context in this sense. The spatial configuration level specifies the expected positions and locations of the objects. Since our interaction likelihoods are based on spatial relationship features, they also capture the spatial context. Finally, the pose context indicates the possible poses of objects with respect to each other. Our horizontality feature roughly encodes the relative surface poses of objects. Hence, our interaction likelihoods also encode the pose context to an extent. As a result, we model the semantic, spatial and pose context of objects in a unified probabilistic framework using spatial relationship feature based interaction likelihoods.

3.2.2 Co-occurrence Based Interaction Likelihood

The co-occurrence based interaction likelihoods are chosen as the baseline approach. For this purpose, we also learn co-occurrence probabilities for the object class pairs and use them as interaction likelihoods. In this case, $p(S_{i,j}|X_i, X_j)$ is computed as

$$p(S_{i,j}|X_i, X_j) = P(\mathbf{C}|X_i, X_j) \tag{3.17}$$

where $P(\mathbf{C}|X_i, X_j)$ represents the probability of co-occurrence of objects having class labels X_i and X_j. Co-occurrence probability does not depend on the instances of object classes. For a given class pair, it is a constant value that is learned from a training set where manual object labels are present:

$$P(\mathbf{C}|X_i = u, X_j = v) = \min\{\mathbf{C}_{u,v}, \mathbf{C}_{v,u}\}, \tag{3.18}$$

$$\mathbf{C}_{u,v} = \frac{(\text{Number of images containing objects of classes } u \text{ and } v) + 1}{(\text{Number of images containing objects of class } v) + 1}, \tag{3.19}$$

$$\mathbf{C}_{v,u} = \frac{(\text{Number of images containing objects of classes } u \text{ and } v) + 1}{(\text{Number of images containing objects of class } u) + 1}, \tag{3.20}$$

$$\mathbf{C}_{u,u} = \frac{(\text{Number of images containing at least 2 objects of class } u) + 1}{(\text{Number of images containing objects of class } u) + 1}. \tag{3.21}$$

Here, $\mathbf{C}_{u,v}$, $\mathbf{C}_{v,u}$ and $\mathbf{C}_{u,u}$ are the co-occurrence probabilities for class pairs.

Note that we increment both numerator and denominator by one in the above equations. By this way, we try to avoid the zero frequency problem. There are two sources for zero counts: impossibility of observation and insufficiency of training samples. In our framework, we assume that every object can co-occur with each other. Thus, the source of zero frequency problem is accepted as insufficiency of training samples. To solve this problem, we prefer adding a pseudocount of one to obtain probabilities other than zero.

Note that as in (3.15), we take minimum of two co-occurrence probabilities in (3.18). By this way, we avoid high probabilities that may come from objects observed in a few images. Suppose that class A object is only present in a single image and class B objects are found in 200 images. Also assume that class A object and one of the class B objects belong to the same image. Then, $\mathbf{C}_{A,B} = (1 + 1)/(200 + 1) \simeq 0.01$ and $\mathbf{C}_{B,A} = (1 + 1)/(1 + 1) = 1.00$. It is clear that choosing the minimum one represents the co-occurrence likelihood better.

We handle this simple contextual information using our interaction likelihood model. Unlike spatial relationship based interaction likelihoods, co-occurrences can only encode the semantic context. In this sense, it is not as strong as the information coming from our relationship features.

Chapter 4

Contextual Agreement Maximization

In Chapters 2 and 3, the sub-components of a scene are described in details. These are the individual objects and the pairwise interactions between them. These sub-components are the building blocks of our contextual scene model. Given these components, the goal of our framework is to decide on the best label for each object by maximizing the contextual agreement between the scene elements. In order to do so, Section 4.1 introduces a scene probability function which is a joint probability defined using object class labels and pairwise object interations. This function is important in the sense that its maximization corresponds to the maximization of contextual agreement between scene elements. Then, Section 4.2 describes the method pursued in maximization of scene probability function. Finally, in Section 4.3, the extendibility of our contextual model is explained.

4.1 Scene Probability Function (SPF)

In this framework, the unknown full 3D model of a scene is approximated in terms of its sub-components, namely the objects and their contextual interactions.

Therefore, we define the scene probability function **SPF** as the joint probability

$$\mathbf{SPF} = P(\mathbf{X}|\mathbf{D},\mathbf{S}) \tag{4.1}$$

of set of variables

$$\mathbf{X} = \{X_1, X_2, \ldots, X_n\} \tag{4.2}$$

that represent the class labels of objects $(X_i, i = 1, \ldots, n)$ computed using

$$\mathbf{D} = \{D_1, D_2, \ldots, D_n\} \tag{4.3}$$

$$\mathbf{S} = \{S_{1,2}, S_{1,3}, \ldots, S_{n-1,n}\} \tag{4.4}$$

that represent the detectors $(D_i, i = 1, \ldots, n)$ and the interactions $(S_{i,j}, i, j = 1, \ldots, n, i \neq j)$ where n is the number of initially detected objects.

The original form of the **SPF** is an unevaluatable joint probability. So, it must be decomposed into computable pieces. For this purpose, we make assumptions:

Assumption 1: The type of the detectors and the interactions are independent given labels of objects.

Assumption 2: Since each object is detected separately, the class label of an object depends only on the type of the detector that recognized it.

Under these assumptions, the **SPF** in (4.1) becomes

$$\begin{aligned}\mathbf{SPF} &= P(\mathbf{X}|\mathbf{D},\mathbf{S}) \\ &= \frac{p(\mathbf{D},\mathbf{S}|\mathbf{X})P(\mathbf{X})}{p(\mathbf{D},\mathbf{S})} \\ &= \frac{P(\mathbf{D}|\mathbf{X})p(\mathbf{S}|\mathbf{X})P(\mathbf{X})}{p(\mathbf{D},\mathbf{S})} \\ &= \frac{\frac{P(\mathbf{X}|\mathbf{D})P(\mathbf{D})}{P(\mathbf{X})}p(\mathbf{S}|\mathbf{X})P(\mathbf{X})}{p(\mathbf{D},\mathbf{S})} \\ &= \frac{P(\mathbf{D})}{p(\mathbf{D},\mathbf{S})}P(\mathbf{X}|\mathbf{D})p(\mathbf{S}|\mathbf{X}).\end{aligned} \tag{4.5}$$

SPF is used for maximization purposes, so the constant term, $\frac{P(\mathbf{D})}{p(\mathbf{D},\mathbf{S})}$, can be discarded. The final form of **SPF** can be written as

$$\begin{aligned}\mathbf{SPF} &= P(\mathbf{X}|\mathbf{D})p(\mathbf{S}|\mathbf{X}) \\ &= \left(\prod_{i=1}^{n} P(X_i|D_i)\right)\left(\prod_{i=1}^{n-1}\prod_{j=i+1}^{n} p(S_{i,j}|X_i,X_j)\right).\end{aligned} \tag{4.6}$$

Terms, $P(X_i|D_i)$ and $p(S_{i,j}|X_i, X_j)$, are already defined in Chapter 2 and Section 3.2, respectively.

In order to maximize the contextual agreement, we must find the values of the variables X_i's $\forall i$ that maximize the **SPF**.

4.2 Maximization of SPF

Maximizing **SPF** is equivalent to maximizing the log-probability function. We can formulate this maximization problem as a binary integer program with an objective function f defined as

$$\begin{aligned} f = &\left(\sum_{i=1}^{n}\sum_{u\in\mathcal{C}_i}\alpha_{iu}\log P(X_i=u|D_i)\right) \\ &+\left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\sum_{\substack{u\in\mathcal{C}_i\\ v\in\mathcal{C}_j}}\beta_{ijuv}\log p(S_{i,j}|X_i=u,X_j=v)\right)\end{aligned} \tag{4.7}$$

where $\mathcal{C}_i$ is the set of class labels that X_i can be assigned to. α_{iu} is an indicator variable for object i being of type u, and β_{ijuv} is an indicator variable for object i being of type u and object j being of type v.

The problem can be written as

$$\begin{array}{lll}
\max & f & \\
\text{s.t.} & \sum_{u\in\mathcal{C}_i}\alpha_{iu}=1, & \forall i \\
 & \sum_{\substack{u\in\mathcal{C}_i\\ v\in\mathcal{C}_j}}\beta_{ijuv}=1, & \forall i<j \\
 & \beta_{ijuv}\le\alpha_{iu}, & \forall i<j, u\in\mathcal{C}_i, v\in\mathcal{C}_j \\
 & \beta_{ijuv}\le\alpha_{jv}, & \forall i<j, u\in\mathcal{C}_i, v\in\mathcal{C}_j \\
 & \alpha_{iu}\in\{0,1\}, & \forall i, u\in\mathcal{C}_i \\
 & \beta_{ijuv}\in\{0,1\}, & \forall i<j, u\in\mathcal{C}_i, v\in\mathcal{C}_j
\end{array} \tag{4.8}$$

Solution of this original formulation is computationally very expensive, due to its NP-hard nature. Fortunately, solving a relaxed version of the problem does not violate the constraints of the original problem. Therefore, we can safely replace $\alpha_{iu}\in\{0,1\}$ with $0\le\alpha_{iu}\le 1$ and $\beta_{ijuv}\in\{0,1\}$ with $0\le\beta_{ijuv}\le 1$. This relaxation is a linear program, so the problem becomes solvable in polynomial time.

Although usually solution gives 0 and 1 for the variables α_{iu} and β_{ijuv} as expected, sometimes intermediate values can be observed. In this case thresholding is applied and variables are assigned to either 0 or 1. If an α_{iu} is equal to 1, it means that among all possible class labels, u is the best choice for object i. This is called *best class label* and denoted as X_i^b. Therefore, when α_{iu} is 1, $X_i^b = u$.

Although our probabilistic model is composed of computable terms, for certain values of X_i, undefined probability values arise. Recall the object class *unknown* defined in Chapter 2. It corresponds to the classes our system does not know about and cannot recognize. Therefore, it is impossible to make observations regarding an *unknown* object. Consequently, the following terms that appear in the body of the objective function are undefined

- $p(S_{i,j}|X_i = unknown, X_j = v)$,
- $p(S_{i,j}|X_i = u, X_j = unknown)$,
- $p(S_{i,j}|X_i = unknown, X_j = unknown)$.

These densities are assumed to be uniform distributions over the spatial relationship feature space. By this way, this undefined densities can be computed and used in our model without any problem.

4.3 Extendibility of the Model

The original formulation of our contextual model aims to determine X_i^b's. However, since it is an extendible model, new unknown variables can easily be integrated. So, best options for other unknowns can be determined besides X_i^b's. We show how our model can be extended to determine the type of spatial relationship between objects in this section.

Let $R_{i,j}$ be the label of spatial relationship between objects i and j. The possible values of $R_{i,j}$ are *on*, *under*, *attached* and etc. In this extended version of the model, we will try to determine the name of the relationship between the objects by finding the *best spatial relationship label*s, $R_{i,j}^b$.

SPF′, the extended scene probability function, is defined as

$$\mathbf{SPF'} = P(\mathbf{X}, \mathbf{R} | \mathbf{D}, \mathbf{S}) \tag{4.9}$$

of sets of variables

$$\mathbf{X} = \{X_1, X_2, \ldots, X_n\} \tag{4.10}$$

$$\mathbf{R} = \{R_{1,2}, R_{1,3}, \ldots, R_{n-1,n}\} \tag{4.11}$$

that represent the class labels of objects $(X_i, i = 1, \ldots, n)$ and their spatial relationships $(R_{i,j}, i, j = 1, \ldots, n, i \neq j)$ computed using

$$\mathbf{D} = \{D_1, D_2, \ldots, D_n\} \tag{4.12}$$

$$\mathbf{S} = \{S_{1,2}, S_{1,3}, \ldots, S_{n-1,n}\} \tag{4.13}$$

that represent the detectors $(D_i, i = 1, \ldots, n)$ and the interactions $(S_{i,j}, i, j = 1, \ldots, n, i \neq j)$ where n is the number of initially detected objects.

Decomposition of **SPF′** into computable terms also requires the assumptions made for **SPF**. In addition, we make one more assumption:

Assumption 3: The spatial relationship label for an object pair depends only on the labels of the participant objects and the interaction between them.

$$\begin{aligned}
\mathbf{SPF}' &= P(\mathbf{X},\mathbf{R}|\mathbf{D},\mathbf{S}) \\
&= P(\mathbf{X}|\mathbf{D},\mathbf{S})P(\mathbf{R}|\mathbf{X},\mathbf{D},\mathbf{S}) \\
&= P(\mathbf{X}|\mathbf{D},\mathbf{S})P(\mathbf{R}|\mathbf{X},\mathbf{S}) \\
&= \frac{p(\mathbf{D},\mathbf{S}|\mathbf{X})P(\mathbf{X})}{p(\mathbf{D},\mathbf{S})}P(\mathbf{R}|\mathbf{X},\mathbf{S}) \\
&= \frac{P(\mathbf{D}|\mathbf{X})p(\mathbf{S}|\mathbf{X})P(\mathbf{X})}{p(\mathbf{D},\mathbf{S})}P(\mathbf{R}|\mathbf{X},\mathbf{S}) \\
&= \frac{\frac{P(\mathbf{X}|\mathbf{D})P(\mathbf{D})}{P(\mathbf{X})}p(\mathbf{S}|\mathbf{X})P(\mathbf{X})}{p(\mathbf{D},\mathbf{S})}P(\mathbf{R}|\mathbf{X},\mathbf{S}) \\
&= \frac{P(\mathbf{D})}{p(\mathbf{D},\mathbf{S})}P(\mathbf{X}|\mathbf{D})p(\mathbf{S}|\mathbf{X})P(\mathbf{R}|\mathbf{X},\mathbf{S}) \\
&= \frac{P(\mathbf{D})}{p(\mathbf{D},\mathbf{S})}P(\mathbf{X}|\mathbf{D})p(\mathbf{S},\mathbf{R}|\mathbf{X}) \\
&= \frac{P(\mathbf{D})}{p(\mathbf{D},\mathbf{S})}P(\mathbf{X}|\mathbf{D})P(\mathbf{R}|\mathbf{X})p(\mathbf{S}|\mathbf{X},\mathbf{R}) \\
&\simeq P(\mathbf{X}|\mathbf{D})P(\mathbf{R}|\mathbf{X})p(\mathbf{S}|\mathbf{X},\mathbf{R}) \\
&= \left(\prod_{i=1}^{n} P(X_i|D_i)\right)\left(\prod_{i=1}^{n-1}\prod_{j=i+1}^{n} P(R_{i,j}|X_i,X_j)p(S_{i,j}|X_i,X_j,R_{i,j})\right)
\end{aligned} \tag{4.14}$$

There are two new terms:

$P(R_{i,j}|X_i,X_j)$**:** Probability of observing relationship $R_{i,j}$ between objects with class labels X_i and X_j. It is estimated from a manually labeled training set as

$$\frac{\text{Number of } (X_i,X_j) \text{ pairs having relationship } R_{i,j}}{\text{Number of } (X_i,X_j) \text{ pairs}} \tag{4.15}$$

$p(S_{i,j}|X_i,X_j,R_{i,j})$**:** Similar to $p(S_{i,j}|X_i,X_j)$, but this distribution is learned using the interactions of object pairs having relationship $R_{i,j}$.

After these definitions, we can introduce the objective function which will be maximized using (4.8) as

$$\begin{aligned} f' = &\left(\sum_{i=1}^{n} \sum_{u \in \mathcal{C}_i} \alpha_{iu} \log P(X_i = u | D_i) \right) \\ &+ \left(\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \sum_{\substack{u \in \mathcal{C}_i \\ v \in \mathcal{C}_j}} \beta_{ijuv} \max_{r \in \mathcal{R}} \log(P(R_{i,j} = r | X_i = u, X_j = v) p(S_{i,j} | X_i = u, X_j = v, R_{i,j} = r)) \right) \end{aligned} \tag{4.16}$$

where $\mathcal{R}$ is the set of possible spatial relationship labels.

After obtaining values of α_{iu} and β_{ijuv}'s as the result of optimization, the determination of the *best* real world spatial relationship label for the object pair (i, j), $R^b_{i,j}$, is straightforward:

$$R^b_{i,j} = \underset{r \in \mathcal{R}}{\operatorname{argmax}} \left(P(R_{i,j} = r | X^b_i, X^b_j) p(S_{i,j} | X^b_i, X^b_j, R_{i,j} = r) \right). \tag{4.17}$$

Similar extensions can be applied on the model by looking at this example. However, since there are not enough groundtruth data and manual labels to train probability models arise in the extended versions, we used the original formulation which just aims to find the best class labels for inidividually detected objects in the experiments.

Chapter 5

Experiments and Evaluation

This chapter is allocated for the experiments we perform to measure the effectiveness of our contextual object recognition framework. First of all, we will describe the data set we use for experimentation in Section 5.1. Then, the object classes of interest and the details regarding the training of the object detectors are presented in Section 5.2. Finally, we will explain the experiments we performed in Sections 5.3 and 5.4, and report the results in Section 5.5.

5.1 Data Sets

We used two data sets for the performance evaluation. The first one is a subset of LabelMe [27] that contains 684 indoor and 1291 outdoor images (1975 images in total). Sample LabelMe images are shown in Figure 5.1. The second one is Bilkent data set that contains 62 indoor and 92 outdoor images (154 images in total). Sample Bilkent images are available in Figure 5.2. Images of both data sets are taken from a large variety of viewing angles and contain more than one object of interest. Thus, these natural scenes are suitable for learning and applying the contextual interaction models.

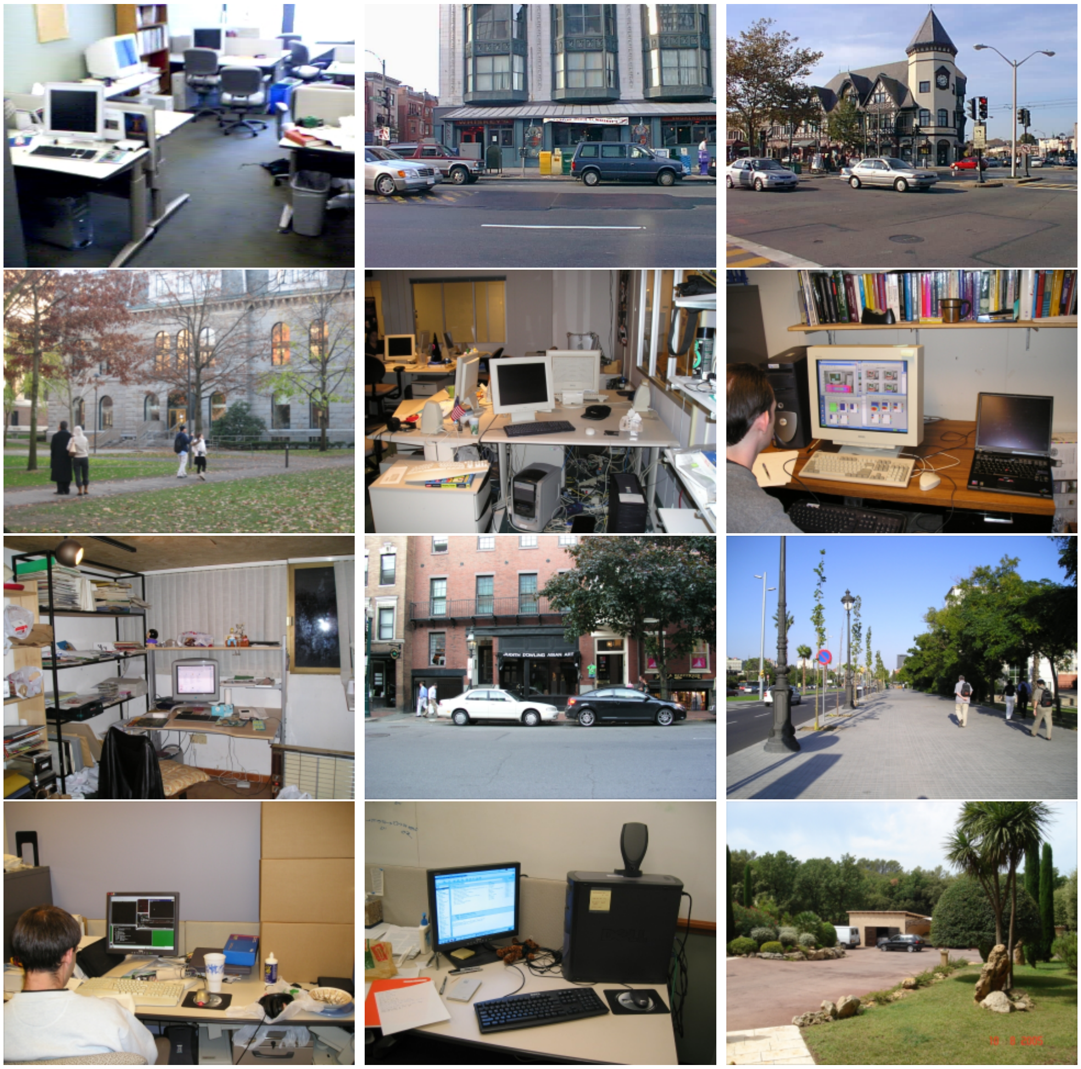


Figure 5.1: Sample images from the LabelMe data set.

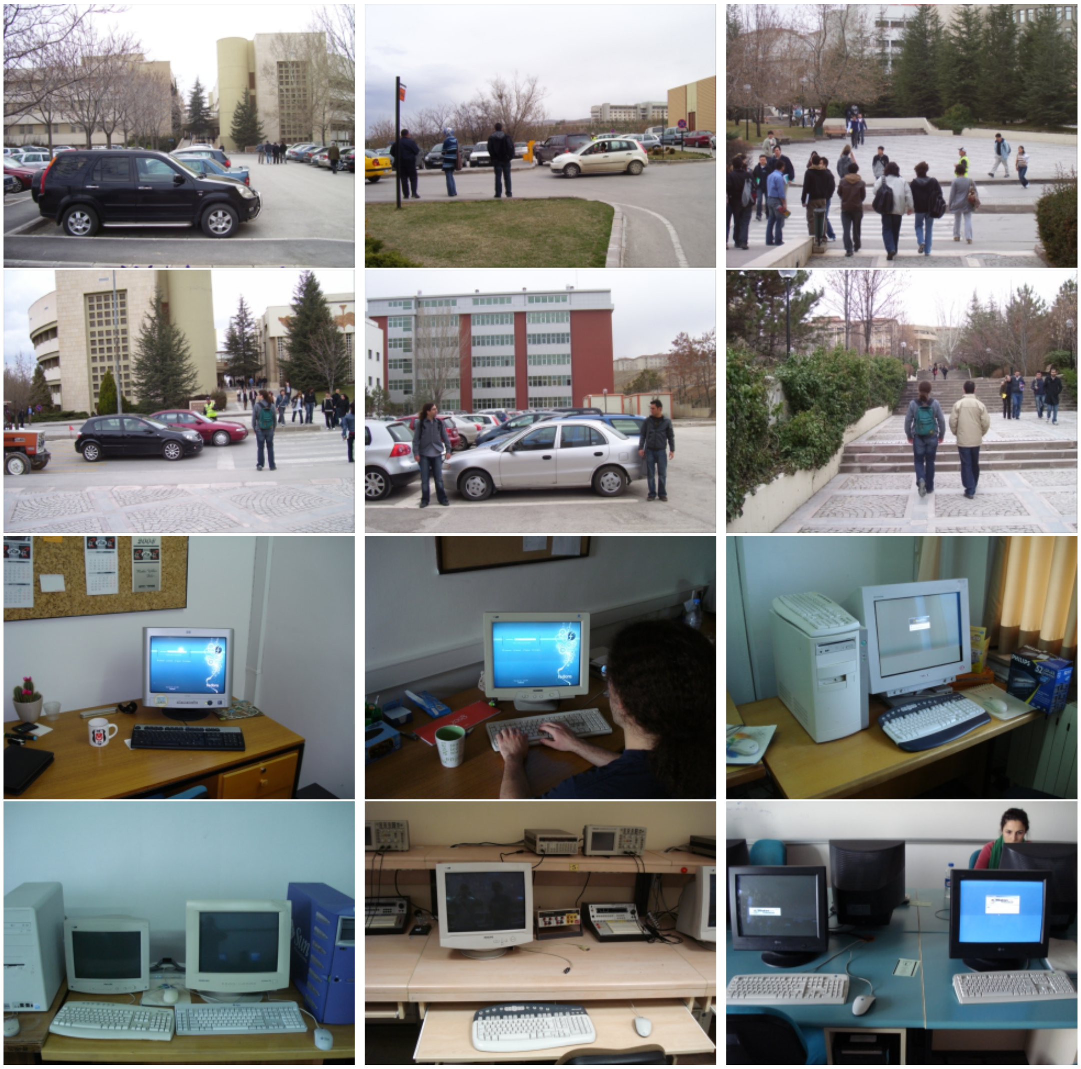


Figure 5.2: Sample images from the Bilkent data set.

5.2 Object Detectors

There are totally 14 object classes we used in the experiments. The categorization of these object classes with respect to their detection methods is presented below:

1. **Shape based object detector with boosting [32]:** computer screen, keyboard, mouse, mug.

2. **Shape based object detector using HOG [7]:** car, person.

3. **Pixel classification based object detector:** sky, tree, grass.

4. **Surface orientation based object detector:** wall, desk, floor, road, building.

The screen, keyboard, mouse, mug and car detectors were trained using manually labeled bounding boxes of the objects in an independent subset of LabelMe. For the person class, we directly used the detector provided in INRIA Object Detection and Localization Toolkit [6]. While sky was detected by a one-class pixel classification based object detector, the tree and grass classes were recognized by a multiple class pixel classification based object detector called *vegetation* detector. The vegetation and sky detectors were trained using manually labeled masks of objects in an image set consisting of the outdoor images of Bilkent University Campus. Note that these images do not belong to the Bilkent data set we used to test our framework. In order to estimate the probability of being a grass or tree object, we computed their frequencies by counting the groundtruth instances found in the training set. These frequencies and the vegetation confidence score were used to estimate the final probabilities of being a grass and tree object as described in Section 2.2. Finally, for surface orientation based object detectors (wall, desk, floor, road and building classes), there was no need for training. As in the grass and tree case, we only computed the frequency of each surface object class. These frequencies were multiplied by the verticality/horizontality confidences to obtain the probability of being a particular surface object as described in Section 2.3.

Table 5.1: The groundtruth object counts in the LabelMe subset with 1975 images.

Class	Count	Class	Count
Building	2390	Mug	143
Car	1295	Person	1176
Desk	658	Road	1048
Floor	95	Screen	906
Grass	910	Sky	963
Keyboard	592	Tree	3262
Mouse	428	Wall	480

5.3 Experiments on the LabelMe Data Set

We performed 5-fold cross validation using the LabelMe subset. Thus, for each validation, 395 independent images were reserved as validation data and remaining 1580 images were allocated for the training. We used the training images to extract the spatial relationship features, estimate co-occurrence probabilities and train the object interaction likelihood models as explained in Section 3.2. Table 5.1 shows the total number of the groundtruth objects found in the LabelMe subset and Table 5.2 shows the distribution of these groundtruth objects in each validation and training data.

There are 18 independent experimental settings. Recall that there are 4 different spatial relationship features we are interested in. These are oriented overlaps, oriented end points ($\theta = 0$), oriented end points ($\theta = 90$) and horizontality. $2^4 - 1 = 15$ of 18 settings utilize all possible combinations of these features as the contextual information using our contextual agreement maximization framework. Following strategies were used in the remaining 3 settings:

- *co*: using co-occurrence probabilities as contextual information in our contextual agreement maximization framework,
- *max*: choosing the class label with the maximum class membership probability (no contextual information, *unknown* class was not used),

Table 5.2: The groundtruth object counts in each LabelMe validation and training data. Column **V.*i*** shows the number of the groundtruth objects for each class of interest (shown in the **Class** column) for the *i*'th validation data. Column **T.*i*** shows the number of the groundtruth objects for each class of interest for the *i*'th training data.

Class	**V.1**	**T.1**	**V.2**	**T.2**	**V.3**	**T.3**	**V.4**	**T.4**	**V.5**	**T.5**
Building	444	1946	543	1847	471	1919	449	1941	483	1907
Car	252	1043	264	1031	253	1042	255	1040	271	1024
Desk	136	522	141	517	129	529	134	524	118	540
Floor	16	79	20	75	19	76	25	70	15	80
Grass	183	727	213	697	172	738	179	731	163	747
Keyboard	112	480	109	483	124	468	129	463	118	452
Mouse	86	342	82	346	89	339	92	336	79	349
Mug	27	116	29	114	30	113	37	106	20	123
Person	208	968	271	905	250	926	196	980	251	925
Road	210	838	208	840	214	834	194	854	222	826
Screen	167	739	195	711	185	721	190	716	169	737
Sky	181	782	189	774	201	762	195	768	197	766
Tree	649	2613	680	2582	646	2612	621	2641	666	2596
Wall	90	390	106	374	86	394	92	388	106	374

- *max/u*: choosing the class label with the maximum class membership probability (no contextual information, *unknown* class was used).

Suppose that {*car* : 0.7, *unknown* : 0.3}, {*grass* : 0.2, *tree* : 0.5, *unknown* : 0.3} and {*wall* : 0.05, *desk* : 0.04, *floor* : 0.05, *road* : 0.01, *building* : 0.15, *unknown* : 0.7} are three objects detected initially in an image where the values after the class names are the class membership probabilities. As we mention above, the *max* approach does not make use of the unknown class. Thus, after using the *max* approach, these three objects are assigned *car*, *tree* and *building* labels, respectively. On the other hand, according to the *max/u* approach considering the *unknown* class, the final labels are *car*, *tree* and *unknown*. Recall that the *unknown* class corresponds to the elimination of the initially detected objects. Thus, the *max/u* approach eliminates the third object. Note that this elimination rule holds for all settings except the *max* approach.

Table 5.3 shows the list of experimental settings and their associated codes for further references.

Table 5.3: Experimental settings and their codes.

Code	**Setting**
O	{oriented overlaps}
E_0	{oriented end points (0)}
E_{90}	{oriented end points (90)}
H	{horizontality}
OE_0	{oriented overlaps, oriented end points (0)}
OE_{90}	{oriented overlaps, oriented end points (90)}
OH	{oriented overlaps, horizontality}
E_0E_{90}	{oriented end points (0), oriented end points (90)}
E_0H	{oriented end points (0), horizontality}
$E_{90}H$	{oriented end points (90), horizontality}
OE_0E_{90}	{oriented overlaps, oriented end points (0), oriented end points (90)}
OE_0H	{oriented overlaps, oriented end points (0), horizontality}
$OE_{90}H$	{oriented overlaps, oriented end points (90), horizontality}
$E_0E_{90}H$	{oriented end points (0), oriented end points (90), horizontality}
$OE_0E_{90}H$	full model
co	co-occurrences
max	choosing the label with maximum probability (*unknown* not used)
max/u	choosing the label with maximum probability (*unknown* used)

Table 5.4: Categorization of 14 classes as indoor or outdoor object.

Indoor Object Classes	**Outdoor Object Classes**
Screen	Car
Keyboard	Person
Mouse	Grass
Mug	Tree
Desk	Sky
Wall	Road
Floor	Building

Table 5.5: The groundtruth object counts in the Bilkent data set with 154 images.

Class	**Count**
Car	41
Grass	45
Keyboard	77
Mouse	64
Mug	32
Person	241
Screen	77

Each validation subset was tested using the 18 settings twice. Firstly, we did not make any assumption regarding the type of the scene. In other words, we conducted the experiments without considering if the input scene is an indoor or outdoor image. Therefore, all detectors were run on all images. Secondly, we only considered the indoor (outdoor) object classes for an indoor (outdoor) scene by running only the detectors for that particular scene type. By this way, we could compare how the additional knowledge of scene type influences contextual object recognition performance. The categorization of 14 classes as indoor and outdoor objects is available in Table 5.4.

5.4 Experiments on the Bilkent Data Set

We also conducted experiments on the Bilkent data set having 154 images. Table 5.5 shows the number of groundtruth object labels found in the Bilkent data set. We used the images of LabelMe subset for training interaction models. Note that all 14 object classes were utilized for contextual agreement maximization, but the performance was measured using 7 classes available as groundtruth. The tests were performed using the same 18 settings shown in Table 5.3. The effect of the scene type information (indoor or outdoor) was again observed by performing experiments on the Bilkent data set twice.

5.5 Results

We use three measures while evaluating the results of the experiments. These are precision, recall and F score. Precision is the fraction of correctly detected objects among all detections. Thus, precision is calculated as

$$p = \frac{\text{number of correct detections}}{\text{total number of detections}}. \tag{5.1}$$

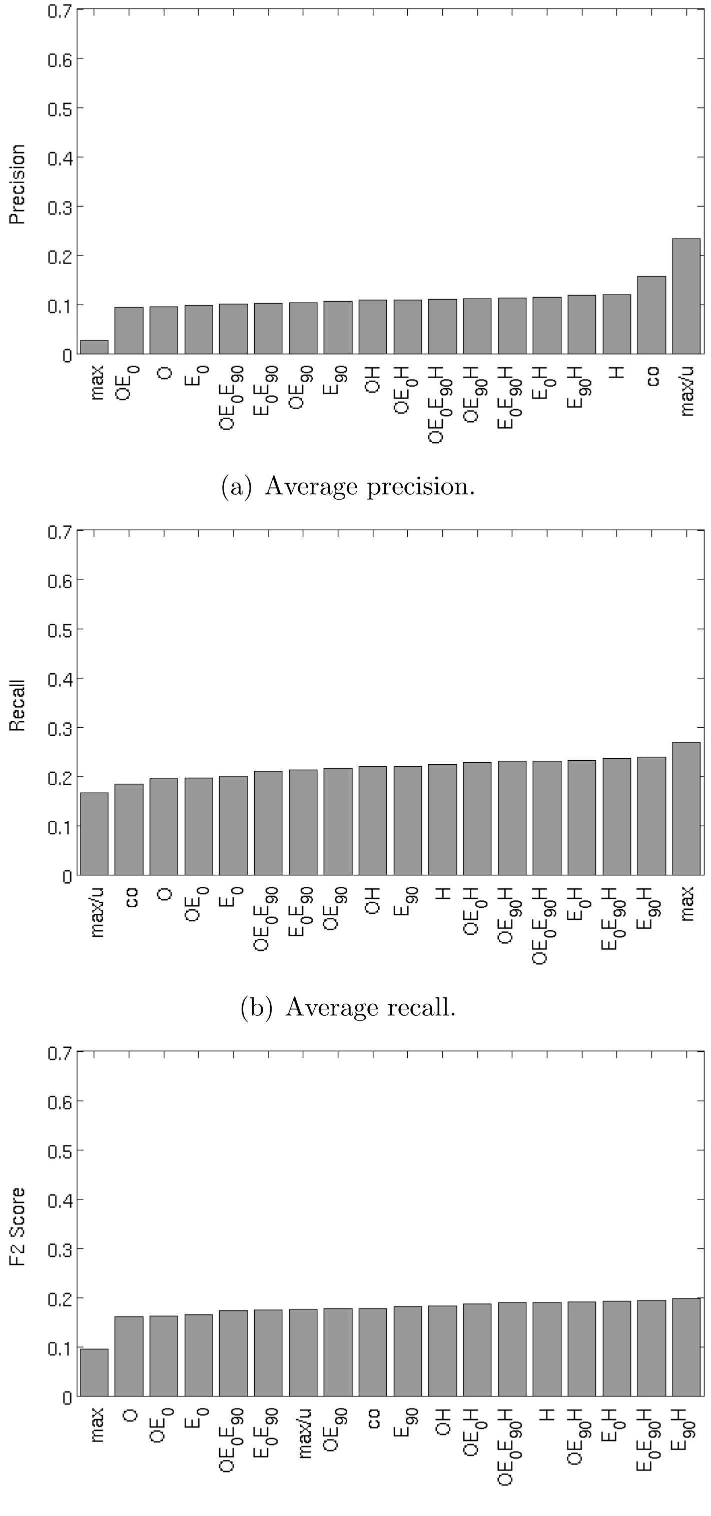
Recall measures the ratio of the correctly detected groundtruth objects as

$$r = \frac{\text{number of correct detections}}{\text{number of groundtruth objects}}. \tag{5.2}$$

Neither precision nor recall alone is sufficient to represent the overall success rate. A recognition system detecting a small number of objects with very high confidence would have high precision but low recall. Another recognition system detecting lots of objects even in a single image would yield high recall but low precision. Thus, precision and recall should both be as high as possible in order to regard a system as successful. For this purpose, another commonly used performance measure called F score is calculated as

$$F_\beta = \frac{(1+\beta^2) \times p \times r}{\beta^2 \times p + r} \tag{5.3}$$

where p is the precision, r is the recall and β is the importance factor of the recall when compared to the precision. We assume that the ratio of the correctly detected groundtruth objects is more important than the proportion of the correct detections among all detections for an object recognition system. In other words, the recall values are assumed to be more significant than the precision values.

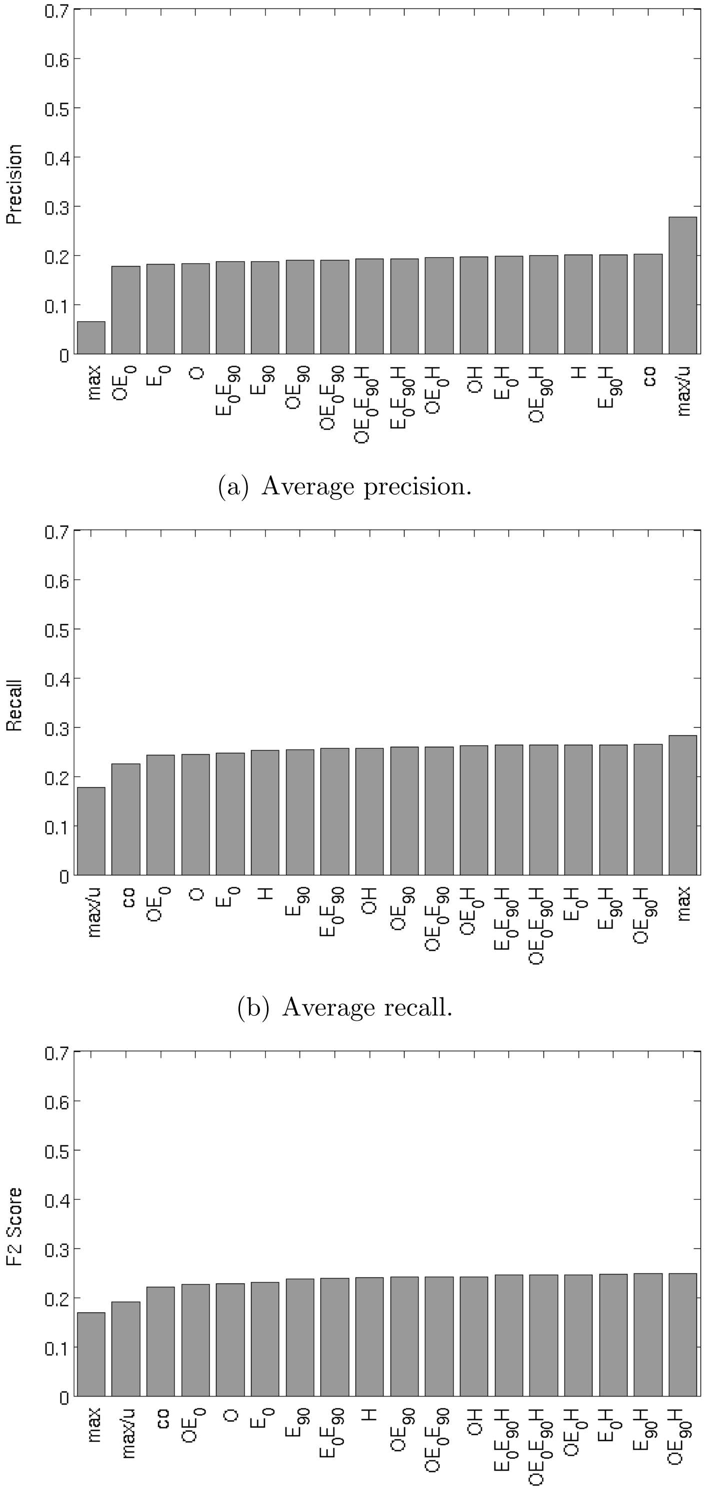


(a) Average precision.

(b) Average recall.

(c) Average F2 score.

Figure 5.3: Average overall performance measurements for 5-fold cross validation applied on the LabelMe data set. The scene type assumption was not used. The settings are sorted in ascending order of the measure used.

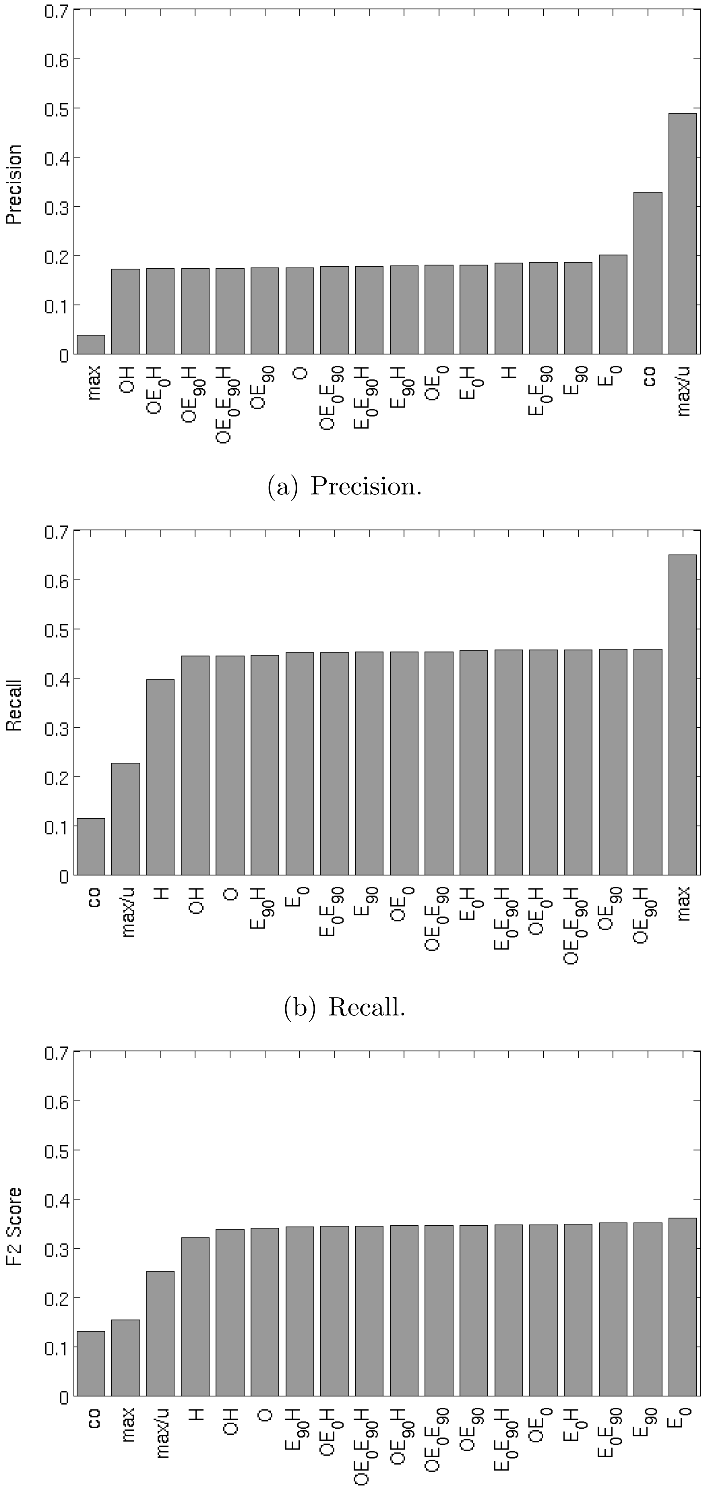


(a) Average precision.

(b) Average recall.

(c) Average F2 score.

Figure 5.4: Average overall performance measurements for 5-fold cross validation applied on the LabelMe data set under the scene type assumption. The settings are sorted in ascending order of the measure used.

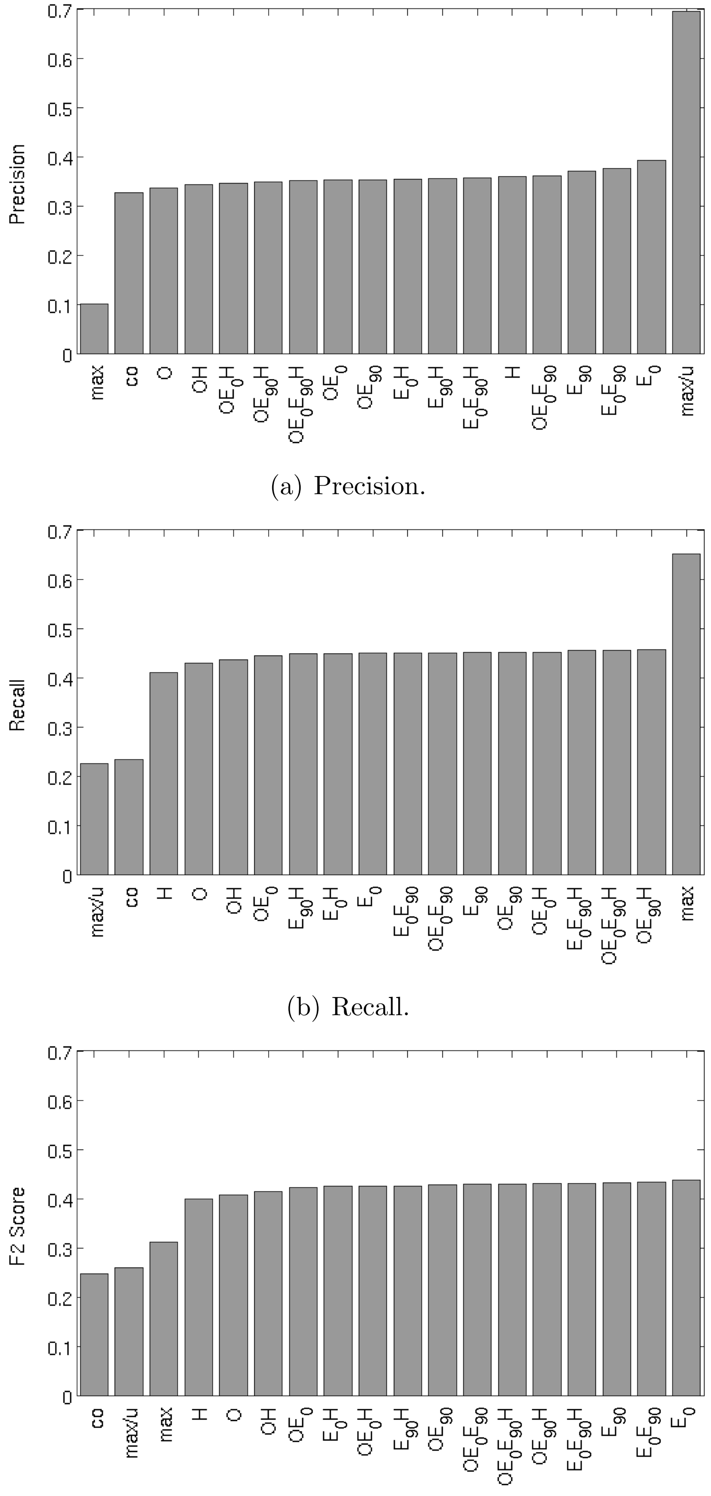


(a) Precision.

(b) Recall.

(c) F2 score.

Figure 5.5: Overall performance measurements for the experiments performed on the Bilkent data set. The scene type assumption was not used. The settings are sorted in ascending order of the measure used.



(a) Precision.

(b) Recall.

(c) F2 score.

Figure 5.6: Overall performance measurements for the experiments performed on the Bilkent data set under the scene type assumption. The settings are sorted in ascending order of the measure used.

Therefore, while computing F scores, we set β as 2 in order to give twice more importance to recall than precision.

Figures 5.3, 5.4, 5.5 and 5.6 present the overall precision, recall and F2 scores for the experiments performed on the LabelMe and Bilkent data sets using the 18 settings. The assumption regarding the scene type was not used for the experiments whose results are shown in Figures 5.3 and 5.5. On the other hand, Figures 5.4 and 5.6 show the results under the scene type assumption.

When only the precision values are considered in all of the figures, the winning approach is to choose the class label with the maximum class membership probability by taking the *unknown* class into account (*max/u*) and the worst approach is to select the class label with the maximum class membership probability without considering the *unknown* class (*max*). This is an expected result. Recall that if an object's label is chosen as *unknown*, this corresponds to its elimination. If the probability of being unknown is greater than the other possible class membership probabilities, the object is removed in the *max/u* approach. Thus, when objects with low detection confidences are totally eliminated, a relatively higher precision is obtained. On the contrary, *max* approach corresponds to directly using the initial detection outputs. This approach is the only one that does not include any object eliminations. This is what makes the precision of *max* approach the lowest.

When only the recall values are considered in each figure, the best approach is found to be *max*. This is reasonable in the sense that other methods may mistakenly remove some of the correctly detected objects during the elimination of the wrong detections. The *max* approach does not have that risk owing to no elimination. By this way, it can keep the recall value high. The winner of the precision values, *max/u*, does not perform well when the recall is considered. This shows that eliminating objects having high probability of being unknown is not the best strategy. This approach increases the probability of removing a correct detection.

It is clear that methods not using any contextual information are only good at either recall or precision. However, a recognition system has to be good at both values. Thus, when F2 scores are considered, neither *max* nor *max/u* approach seems promising. Our contextual agreement maximization framework using spatial relationship features as object interactions is the best approach in terms of F2 scores. Although in Figure 5.3(c), *co* and *max/u* approaches seem better than some of spatial relationship based interaction models, all of our models outperform *max*, *max/u* and *co* in Figures 5.4(c), 5.5(c) and 5.6(c).

When the scene type information is utilized, the precision, recall and F2 score values are greater than the values observed in the cases of no additional information. The precision increases, because a detector of an indoor (outdoor) object is not run for the outdoor (indoor) images. Therefore, the number of possible wrong detections decreases which causes an increase in the precision. The recall remains constant for *max* and *max/u* approaches. However, the recall increases for *co* and spatial relationship based object interaction models. This is because the maximization of contextual agreement in the scene is more meaningful for the indoor (outdoor) objects using the indoor (outdoor) context. Suppose that one screen, one keyboard, one car and one road are detected in a scene. It is apparent that some of these objects contradict with the scene context. When the scene type is unknown, the possibility of keeping the wrong objects and eliminating the correct ones is higher. Therefore, in order to have better recall values for a contextual object recognition system, the additional information of the scene type should be taken into account.

The experiments on the LabelMe data set show that performance using the settings including the horizontality feature is relatively higher. However, this feature is not significant in the results of the Bilkent data set based experiments. When both data sets are considered, it is clear that the best performing features are the oriented end points features. On the other hand, the oriented overlaps demonstrate an average success in each experiment. These interpretations lead to the fact that there is no best feature combination to be used as contextual interactions. The best combination depends on the object classes of interest used in the experiments.

The F2 scores of our context based baseline approach, *co*, are lower when compared to the spatial relationship based methods. This shows that using the spatial (oriented overlaps and oriented end points) and pose context (horizontality) in addition to the semantic context (co-occurrences) results in better recognition performance.

Tables 5.6, 5.7, 5.8 and 5.9 show F2 scores for each object class used in the experiments on the LabelMe and Bilkent data sets. Recall that shape based object detectors using HOG (car and person detectors) do not tend to report false detections. Thus, their precision values are high when compared to their recall values. So, their precision cannot be improved drastically using the methods that can eliminate some of the initial detections. Consequently, F2 score cannot be also improved. This makes the *max* approach the best option for HOG based detectors. The objects like car and person are the reliable sources of the scene context since the detection confidence scores are high in these detectors.

Boosting based detectors are different in the sense that they are low precision but high recall recognizers. Thus, using *max/u* approach seems to be the best option in some cases. For example, *max/u* performance is highest for the classes like keyboard, mouse and mug with very low precision in Table 5.6. The reason is that the context based methods are not always as good as *max/u* approach at removing objects with low class membership probabilities. Only the screen class whose shape is more discriminative than the others can be detected with higher precision. Thus, context based methods perform better for the screen class.

Results show that the F2 scores of pixel classification and surface orientation based detectors are improved by the contextual models. Since the frequency of the tree objects in the training sets is greater than the frequency of the grass objects, a vegetation object can only be initially labeled as a tree. Therefore, the *max* and *max/u* approaches cannot detect a grass object. However, by the help of the scene context, the disambiguation of the grass and tree becomes possible. Thus, F2 scores of the grass class is greater than 0 under context based experiment settings. The same situation is valid for the surface object classes, building, desk, floor, road and wall. These frequency based objects can

Table 5.6: Average F2 scores for each object class used in the experiments on the LabelMe data set. These experiments were performed without using the scene type information. The *italic* values are the lowest of their rows and the **bold** ones are the highest.

		Setting						
		max	max/u	co	O	$\mathrm{E}_0\mathrm{E}_{90}$	H	$\mathrm{OE}_0\mathrm{E}_{90}\mathrm{H}$
Class	building	0.25	**0.26**	0.25	*0.11*	0.17	0.18	0.19
	car	**0.17**	*0.02*	0.08	0.07	0.08	0.08	0.08
	desk	*0.00*	*0.00*	0.01	0.15	**0.16**	0.10	0.15
	floor	*0.00*	*0.00*	*0.00*	0.09	*0.00*	0.07	**0.12**
	grass	*0.00*	*0.00*	*0.00*	0.09	0.06	**0.15**	0.13
	keyboard	*0.07*	**0.39**	0.17	0.32	0.32	0.31	0.32
	mouse	*0.01*	**0.12**	0.07	0.06	0.06	0.06	0.06
	mug	*0.01*	**0.13**	0.03	0.07	0.07	0.07	0.07
	person	**0.08**	0.01	*0.00*	0.05	0.05	0.04	0.05
	road	0.50	0.52	0.51	0.38	*0.36*	**0.54**	0.48
	screen	0.42	0.38	*0.15*	**0.52**	0.52	0.51	0.52
	sky	*0.32*	0.32	0.32	0.33	0.32	**0.36**	0.33
	tree	0.14	*0.01*	0.15	0.13	0.15	0.15	**0.15**
	wall	*0.00*	*0.00*	*0.00*	0.03	0.01	**0.03**	0.02

Table 5.7: Average F2 scores for each object class used in the experiments on the LabelMe data set. These experiments were performed using the scene type information. The *italic* values are the lowest of their rows and the **bold** ones are the highest.

		Setting						
		max	max/u	co	O	$\mathrm{E}_0\mathrm{E}_{90}$	H	$\mathrm{OE}_0\mathrm{E}_{90}\mathrm{H}$
Class	building	0.30	0.30	0.31	*0.29*	0.31	**0.32**	0.32
	car	**0.17**	*0.02*	0.08	0.07	0.08	0.08	0.08
	desk	0.18	0.18	0.15	0.22	0.22	*0.12*	**0.22**
	floor	*0.00*	*0.00*	*0.00*	0.11	*0.00*	**0.19**	0.11
	grass	*0.00*	*0.00*	*0.00*	0.10	0.06	**0.17**	0.16
	keyboard	*0.18*	0.42	0.40	0.46	**0.46**	0.41	**0.46**
	mouse	*0.04*	**0.14**	0.12	0.11	0.11	0.10	0.11
	mug	*0.03*	**0.16**	0.06	0.15	0.15	0.14	0.15
	person	**0.08**	0.01	*0.00*	0.04	0.05	0.04	0.05
	road	0.55	**0.57**	0.56	*0.52*	0.55	0.56	0.53
	screen	0.53	*0.38*	0.41	0.54	**0.54**	0.52	0.54
	sky	0.35	*0.35*	0.35	0.35	0.35	**0.39**	0.35
	tree	0.17	*0.01*	0.17	0.14	**0.17**	0.17	0.17
	wall	0.04	0.04	**0.05**	0.03	*0.01*	0.05	0.02

Table 5.8: F2 scores for each object class used in the experiments on the Bilkent data set. These experiments were performed without using the scene type information. The *italic* values are the lowest of their rows and the **bold** ones are the highest.

		Setting						
		max	max/u	co	O	E_0E_{90}	H	$OE_0E_{90}H$
Class	car	**0.39**	*0.03*	0.26	0.23	0.26	0.26	0.26
	grass	*0.00*	*0.00*	*0.00*	0.03	*0.00*	**0.04**	0.04
	keyboard	*0.15*	**0.72**	0.32	0.59	0.59	0.52	0.59
	mouse	*0.05*	0.17	0.11	**0.18**	0.18	0.15	0.18
	mug	0.05	0.13	*0.00*	**0.20**	0.20	0.18	0.20
	person	**0.64**	0.13	*0.02*	0.39	0.41	0.37	0.41
	screen	0.60	0.44	*0.34*	**0.66**	**0.66**	0.66	**0.66**

Table 5.9: F2 scores for each object class used in the experiments on the Bilkent data set. These experiments were performed using the scene type information. The *italic* values are the lowest of their rows and the **bold** ones are the highest.

		Setting						
		max	max/u	co	O	E_0E_{90}	H	$OE_0E_{90}H$
Class	car	**0.42**	*0.03*	0.26	0.23	0.26	0.26	0.26
	grass	*0.00*	*0.00*	*0.00*	0.04	*0.00*	**0.06**	0.06
	keyboard	*0.35*	**0.74**	0.64	0.74	0.74	0.67	0.74
	mouse	*0.12*	0.20	0.27	**0.30**	0.30	0.27	0.30
	mug	0.13	0.15	*0.00*	**0.35**	**0.35**	0.32	**0.35**
	person	**0.65**	0.13	*0.01*	0.35	0.41	0.37	0.41
	screen	**0.73**	*0.43*	0.58	0.69	0.69	0.68	0.69

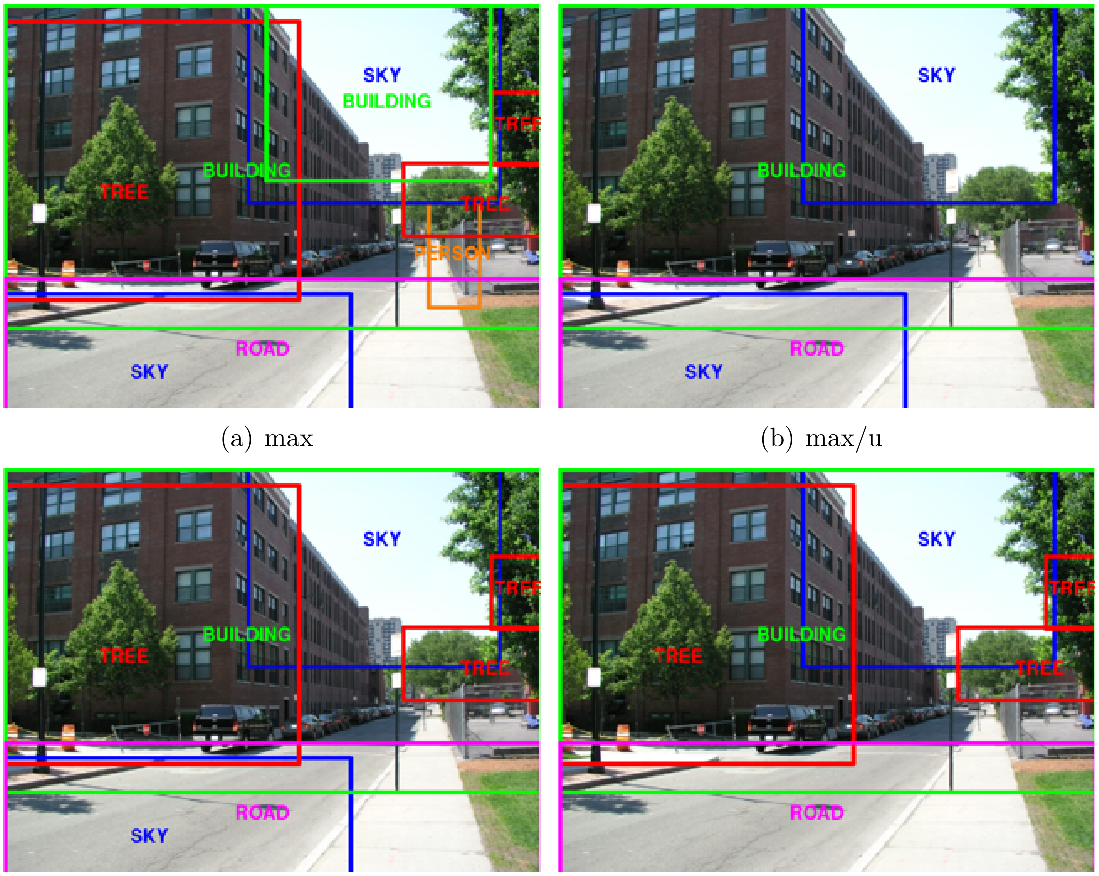
also be disambiguated best under the contextual information. Recall that it is very difficult to detect surface objects using traditional detectors that use color, texture or shape features. On the other hand, it is clear that the detection of such objects becomes possible by utilizing the surface orientations in our contextual framework.

Figures 5.7, 5.8, 5.9 and 5.10 show the sample final label assignments for the sample input images from both LabelMe and Bilkent data sets using the *max*, *max/u*, *co* and the best performing spatial relationship feature set settings. Note that we show examples from the experiments in which the scene type information was utilized. The detection masks are shown as bounding boxes in the figures in order to avoid the possible clutter.

The sky object overlapping with the road object could only be eliminated by our framework using the $OE_{90}H$ setting as shown in Figure 5.7. Similarly, the final label for the vegetation object located below the leftmost tree could only be changed from tree to grass class again under the $OE_{90}H$ setting in Figure 5.8. The screen and the keyboard could only be detected concurrently under the E_0 setting as shown in Figure 5.9. Likewise, in Figure 5.10, the mug in addition to the screen and the keyboard was correctly reported as a final detection when the E_0 setting was used.

Although there are some flaws in the final object detections observed under the settings utilizing our spatial relationship based contextual interaction models, they still yield the most reasonable results when compared to other approaches.

Besides precision, recall and F2 score based performance evaluation, the computational time and complexity analysis of our contextual agreement maximization framework is also important. As we mentioned in Section 4.2, our scene probability function is maximized using linear optimization. We used MATLAB's Optimization Toolbox [34] that utilizes a primal-dual interior-point method to solve a linear program in polynomial time. During the experiments, the optimization for each input image was always terminated when an optimum solution was obtained. Optimization for an image with a few (≤ 20) initially detected objects took less than 1 second. When the number of the initially detected objects



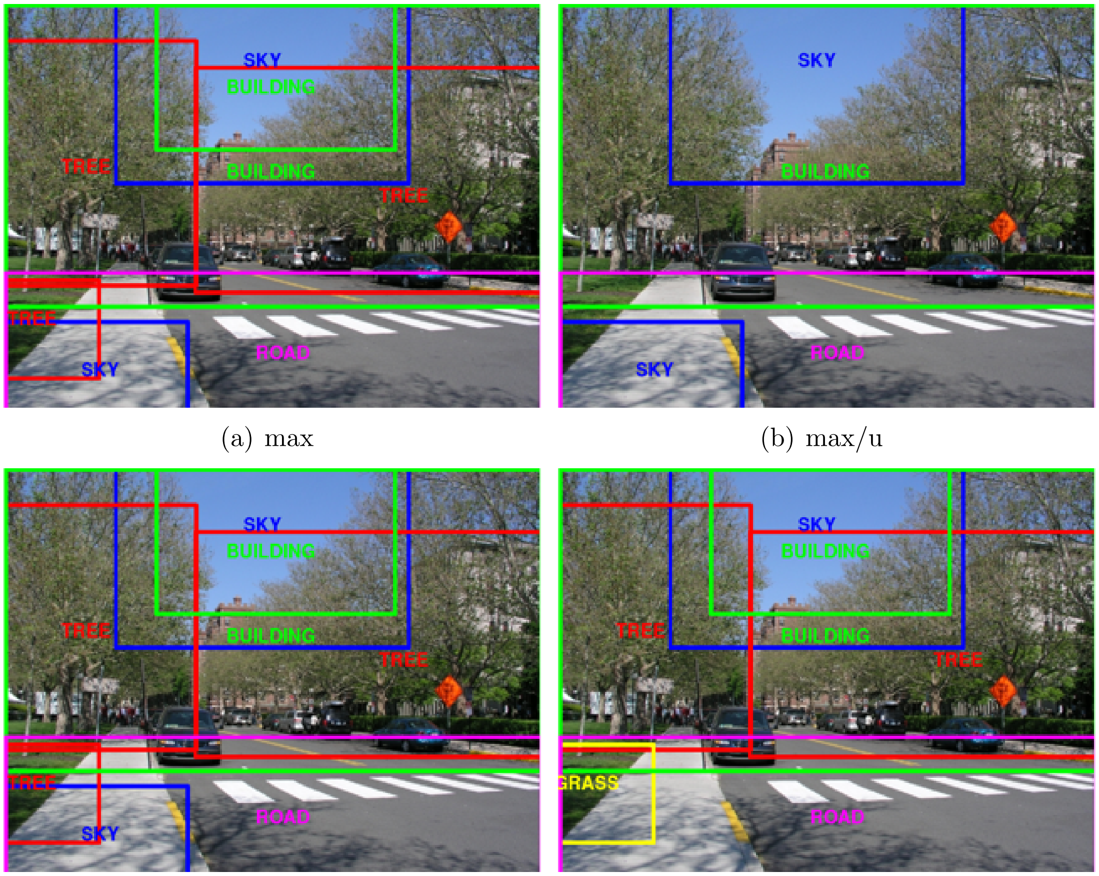
(a) max

(b) max/u

(c) co

(d) $\mathrm{OE}_{90}\mathrm{H}$

Figure 5.7: Sample final label assignments using the *max*, *max/u*, *co* and $\mathrm{OE}_{90}\mathrm{H}$ (the best performing feature set) settings for an image from the LabelMe data set.



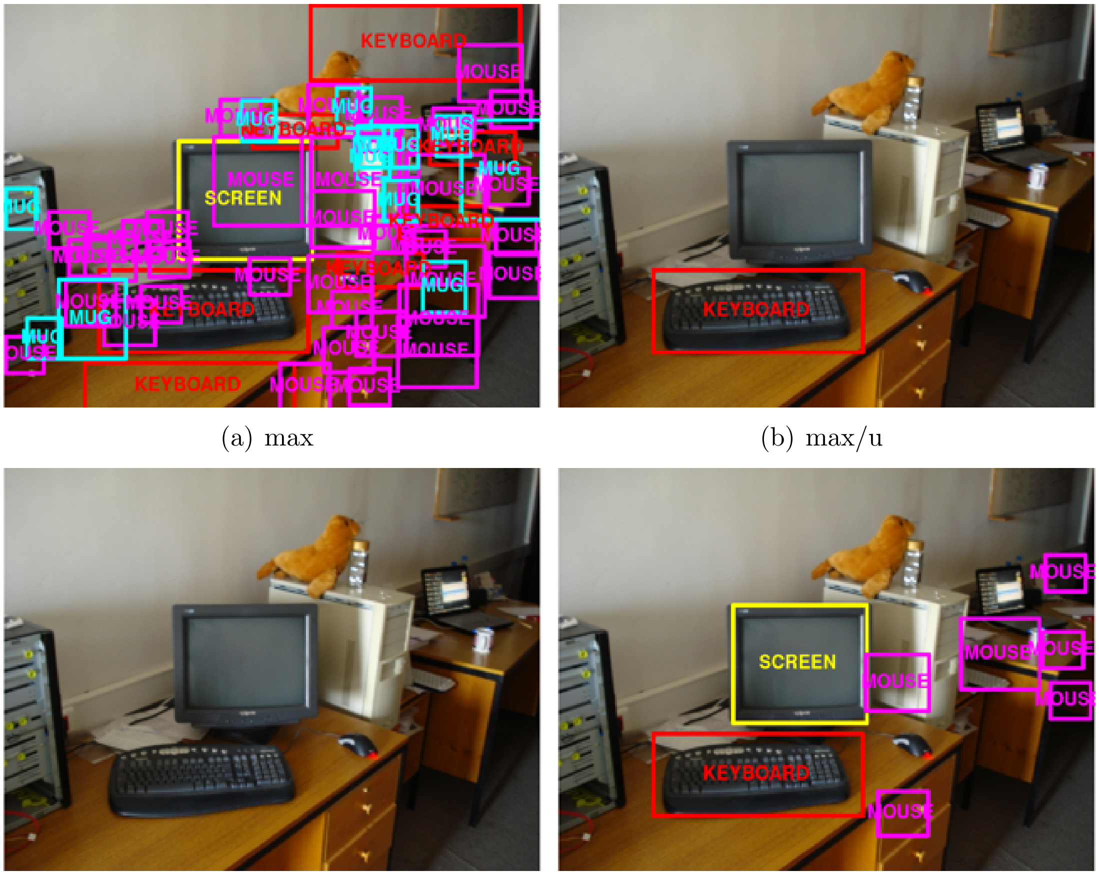
(a) max

(b) max/u

(c) co

(d) $\mathrm{OE}_{90}\mathrm{H}$

Figure 5.8: Sample final label assignments using the *max*, *max/u*, *co* and $\mathrm{OE}_{90}\mathrm{H}$ (the best performing feature set) settings for an image from the LabelMe data set.



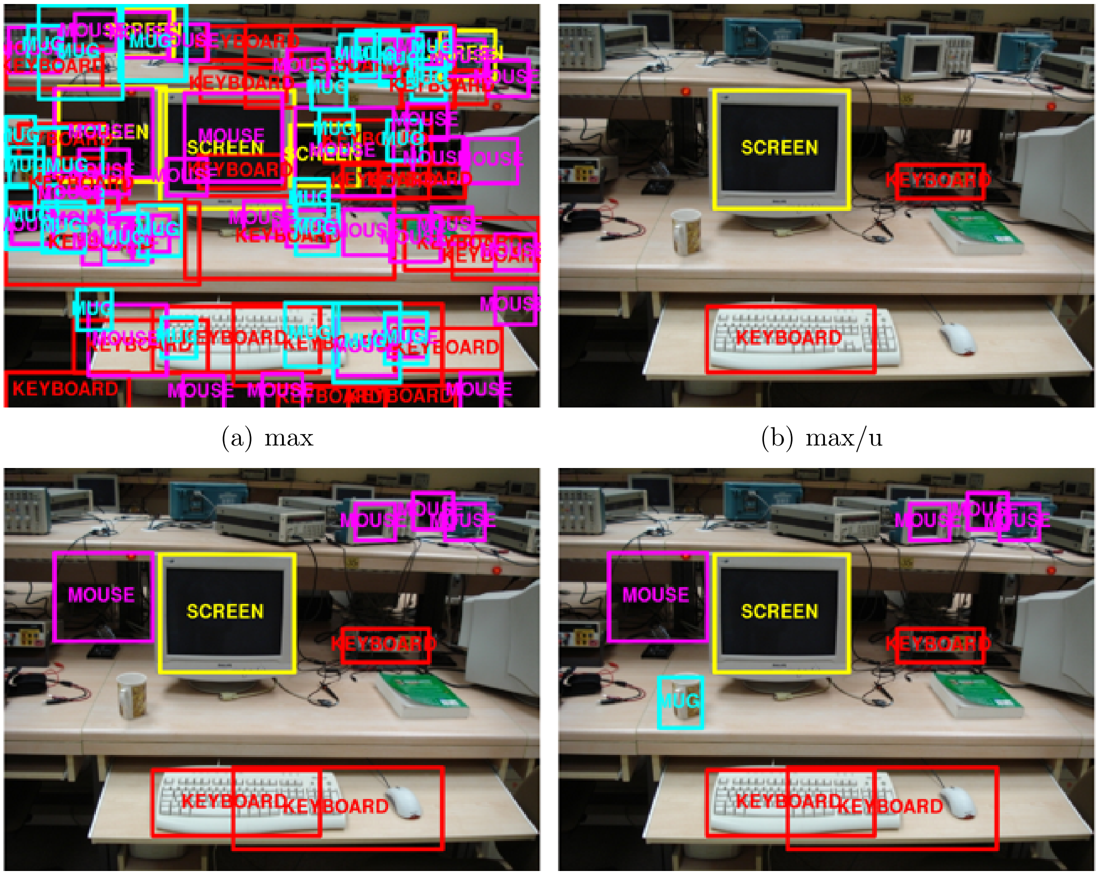
(a) max

(b) max/u

(c) co

(d) E_0

Figure 5.9: Sample final label assignments using the *max*, *max/u*, *co* and E_0 (the best performing feature set) settings for an image from the Bilkent data set.



(a) max

(b) max/u

(c) co

(d) E_0

Figure 5.10: Sample final label assignments using the *max*, *max/u*, *co* and E_0 (the best performing feature set) settings for an image from the Bilkent data set.

was close to 110, the time taken was approximately 5 minutes. The other context based approaches in the literature can handle a few number of candidate objects (initially detected objects) in a relatively longer time. For example, Rabinovich et al. [26] obtain the candidate objects using a segmentation algorithm which yields at most 10 segments. They report that the application of the contextual constraints on a given segmentation takes up to 7 seconds. When 10 candidate objects in 7 seconds is compared to 20 objects in 1 second, our framework outperforms [26] that utilizes the widely used conditional random field (CRF) framework to implement the contextual inference mechanism. Note that it is also not feasable to handle 110 candidate objects (variables) in a CRF framework. In this sense, our framework is more practical than the CRF based methods found in the literature.

Chapter 6

Conclusions

In this thesis, we described a contextual object recognition framework dedicated to increase the overall recognition performance. First, all object detectors incorporated into our framework are run on the input still image. The initial object detections with the class membership probabilities are obtained at the end of this procedure. Next, the contextual interactions among these candidate objects are estimated. The baseline interaction used in our framework is co-occurrence probabilities that encode only the semantic context of a scene. On the other hand, our proposed interactions are based on three different spatial relationship features. The oriented overlaps feature captures the relative overlap amounts in different orientations. The oriented end points feature encodes the relative positions of the objects using their projections to an orientation of interest. These two orientation based spatial relationship features convey information regarding the relative scales, positions and locations of the objects which constitute the spatial context of a scene. There is also a third feature called horizontality. It captures the relative horizontality of two objects that is a rough representation of the pose context. Then, an object pair's interaction likelihood based on our spatial relationship features is calculated as a smoothed histogram estimate. Next, the initial object detections and the pairwise contextual interaction likelihoods are utilized to obtain the best scene configuration using our contextual agreement maximization framework. Finding the best scene configuration corresponds to the elimination of the objects that are inconsistent with the scene context and the disambiguation of the class labels recognized by the multiple class object detectors. In order to implement the elimination mechanism, we employed an extra object class called *unknown* (the case of not being able to call an object a member of the classes recognized by our system). This framework maximizes a novel scene probability function that is defined jointly using both the individual object labels and their pairwise contextual interactions. This maximization problem is solved using linear optimization.

We performed experiments on two different data, the LabelMe [27] and Bilkent data sets whose images are taken from a large variety of viewing angles and contain more than one object of interest. Hence, these natural scenes were suitable for learning and applying the contextual interaction models. Experimental results show that the best strategy in an object recognition system is using the contextual models when the overall F2 scores are considered. Among the context based approaches, spatial relationship based object interactions outperform the co-occurrence based interactions. This shows that how the spatial and pose context are important besides the semantic context. We also investigated which combination of the spatial relationship features performs best in the contextual agreement maximization. Results show that the best combination depends on the objects contributing to the overall scene context. Thus, there is no fixed set of features to be used in our framework. In addition, the results indicate that the additional information about the scene type (indoor and outdoor) causes an increase in the F2 scores of the context based approaches.

Besides the overall performance, we also examined how different object classes behave under different experiment settings. The results show that F2 scores of the object detectors with high precision relative to their low recall cannot be improved using the contextual information. Instead, they can be used as a reliable source of context. On the other hand, the detectors with high recall relative to their low precision can be improved best by choosing the class label with the maximum class membership probability and eliminating the candidate objects having low detection confidence scores. However, for pixel classification and surface orientation based object detectors, it is shown that the best strategy is using the spatial relationship based context models. The recognition is only possible by the contextual agreement maximization for the objects whose detections depend on their frequencies in a training set.

This contextual object recognition system is an extendible framework as explained in Section 4.3. New unknowns can easily be incorporated into our scene probability function. Hence, the extended version of the framework together with more object detectors may be used to build a large scale computer vision system in the future. Our current framework is observed to be able to efficiently handle approximately 110 initially detected objects belonging to 14 classes in the experiments. Note that the number of classes would be greater than 14 in a larger scale system. This would lead to thousands of initially detected objects to be handled. Then, using the generic linear optimization would not be tractable any more. Thus, one of our future work is to devise a more efficient algorithm to solve the maximization problem.

Recall that the contextual agreement maximization using the spatial relationship feature combinations including the horizontality feature outperform the other approaches in the experiments performed on the LabelMe data set (Figures 5.3(c) and 5.4(c)). Thus, we will focus on the spatial relationship features that can model the pose context in a more sophisticated way as another future work. The overall recognition performance may be improved more by incorporating such features into our contextual interaction model.

Bibliography

[1] S. Aksoy, K. Koperski, C. Tusk, G. Marchisio, and J. Tilton. Learning Bayesian classifiers for scene classification with a visual grammar. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):581–589, 2005.

[2] I. Biederman, R. Mezzanotte, and J. Rabinowitz. Scene perception: detecting and judging objects undergoing relational violations. *Cognitive Psychology*, 14(2):143, 1982.

[3] N. Campbell, B. Thomas, and T. Troscianko. Automatic segmentation and classification of outdoor images using neural networks. *IJNS*, 8(1):137, 1997.

[4] P. Carbonetto, O. De Freitas, and K. Barnard. A Statistical model for general contextual object recognition. In *ECCV*, pages 350–362, 2004.

[5] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

[6] N. Dalal. INRIA object detection and localization toolkit. http://pascal.inrialpes.fr/soft/olt/.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[8] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *CVIU*, 106(1):59–70, 2007.

[9] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, 2005.

[10] R. Fergus. A simple parts and structure object detector. ICCV Short Courses on Recognizing and Learning Object Categories, 2005.

[11] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(1):36–51, 2008.

[12] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 100(22):67–92, 1973.

[13] C. Galleguillos, A. Rabinovich, and S. Belongie. Object categorization using co-occurrence, location and appearance. In *CVPR*, 2008.

[14] G. Heitz and D. Koller. Learning spatial context: using stuff to find things. In *ECCV*, 2008.

[15] H. Hock, G. Gordon, and R. Whitehurst. Contextual relations: the influence of familiarity, physical plausibility, and belongingness. *Percept Psychophys*, 16:4–8, 1974.

[16] D. Hoiem, A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 75(1):151–172, 2007.

[17] F. Kalaycılar and S. Aksoy. Object detection with contextual inference. In *Signal Processing and Communications Applications Conference*, 2009.

[18] S. Kumar and M. Hebert. A hierarchical field framework for unified context-based classification. In *ICCV*, 2005.

[19] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[20] P. Lipson, E. Grimson, and P. Sinha. Configuration based scene classification and image indexing. In *CVPR*, 1997.

[21] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[22] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

[23] A. Oliva and A. Torralba. The role of context in object recognition. *Trends in Cognitive Sciences*, 11(12):520–527, 2007.

[24] C. Pantofaru, C. Schmid, and M. Hebert. Object recognition by integrating multiple image segmentations. In *ECCV*, 2008.

[25] D. Parikh, L. Zitnick, and T. Chen. From appearance to context-based recognition: dense labeling in small images. In *CVPR*, 2008.

[26] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *ICCV*, 2007.

[27] B. Russell, A. Torralba, K. Murphy, and W. Freeman. LabelMe: A database and web-based tool for image annotation. *IJCV*, 77(1):157–173, 2008.

[28] H. Schneiderman and T. Kanade. Object detection using the statistics of parts. *IJCV*, 56(3):151–177, 2004.

[29] A. Singhal, J. Luo, and W. Zhu. Probabilistic spatial context models for scene content understanding. In *CVPR*, 2003.

[30] D. Tax. *One-class classification; Concept-learning in the absence of counter-examples.* PhD thesis, Delft University of Technology, 2001.

[31] A. Torralba. Contextual priming for object detection. *IJCV*, 53(2):169–191, 2003.

[32] A. Torralba. A simple object detector with boosting. ICCV Short Courses on Recognizing and Learning Object Categories, 2005.

[33] A. Torralba, K. Murphy, W. Freeman, and M. Rubin. Context-based vision system for place and object recognition. In *ICCV*, 2003.

[34] Y. Zhang. Solving large-scale linear programs by interior-point methods under the MATLAB environment. *Optimization Methods and Software*, 10(1):1–31, 1998.