UNIVERSITÄT HAMBURG

# A Generic Middle Layer for Image Understanding

Kasim Terzić

Doktorarbeit

Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Informatik

Genehmigung der MIN Fakultät, Fachbereich Informatik der Universität Hamburg im Auftrag von:

| | |
|---|---|
| Erstgutachter: | Prof. Bernd Neumann |
| Zweitgutachter: | Prof. Jianwei Zhang |
| Vorsitzender des Prüfungsausschusses: | Prof. Tilo Böhmann |

Tag der Disputation: 11.07.2011

# Abstract

Although humans can understand complex scenes with apparent ease, the problem remains difficult for artificial scene interpretation systems. A central problem facing such systems is reliable detection of objects present in a scene from raw image data. This thesis presents a probabilistic middle-layer architecture which can make use of high-level scene context to improve the detection and recognition of scene objects. The main features of the proposed middle layer are a novel contextual classification scheme, an intermediate-level representation of the scene in terms of object views, and accurate modelling of detection certainty. The architecture is generic and can be applied to any low-level algorithm which delivers class probabilities and any high-level reasoning system which implements the needed interface. The performance is evaluated on images from the façade domain using a wide range of low-level detectors and several high-level interpretation systems.

# Zusammenfassung

Das Verstehen von komplexen Szenen ist eine leichte Aufgabe für Menschen, bleibt aber nach wie vor schwierig für künstliche Szeneninterpretationssysteme. Dabei ist die zuverlässige Erkennung von Szenenobjekten in Bilddaten ein zentrales Problem. Diese Arbeit stellt eine probabilistische Architektur vor, die wissensbasierten Szenenkontext benutzen kann, um die Detektion und Erkennung von Szenenobjekten zu verbessern. Sie funktioniert dabei als Schnitstelle zwischen der Bildverarbeitungsalgorithmen und höheren Szenenmodellen. Die Hauptmerkmale der vorgestellten Architektur sind ein neuer kontextbasierter Klassifikationsmechanismus, eine Zwischenrepräsentation der Szene basierend auf *Object Views* und genaue Modellierung der Detektionsungenauigkeit. Die Architektur ist generisch aufgebaut und kann durch generische Schnittstellen mit verschiedenen probabilistischen Bildverarbeitungsalgorithmen kombiniert werden, sowie mit verschiedenen Systemen für höhere Bilddeutung. Die Evaluierung der Architektur in der Kombination mit verschiedenen Bildverarbeitungs- und Bilddeutungsmodulen wurde auf Bilddaten aus der Fassadendomäne durchgeführt.

# Acknowledgments

Work as complex as scene interpretation is often based on good teamwork. Some of the work presented in this thesis was performed in close cooperation with my colleagues. First of all, I would like to thank my supervisor, Prof. Bernd Neumann, for the support and guidance during my time in Hamburg and many ideas which helped shape this work.

The crisp high-level interpretation system based on the KONWERK tool was written by Dr. Lothar Hotz. He was involved in all the experiments involving crisp interpretation (Sec. 5.2) and helped with the design of the IPC interface to the high level system (Sec. 4.2.1) and the OWL format (Sec. 4.2.4). The work on the Bayesian Compositional Hierarchy was done together with Arne Kreutzmann. This includes all the experiments related to context-based classification (Sec. 5.4.2). The incremental learning experiment was done together with Johannes Hartz (Sec. 5.2.1).

The low-level image processing modules were provided by partners from the eTRIMS project. Mohammad Jahangiri provided the interest region detector and helped with the experiments using local context based on Markov Random Fields (Sec. 5.4.2.2). Jan Šochman provided the trainable AdaBoost-based object detector and helped with the experiments involving visual structures (Sec. 5.2.2). Martin Drauschke provided the stable region detector used in the segmentation experiment (Sec. 5.4.3). I would also like to thank all my colleagues for numerous discussions.

This work was supported by the European Community under the grants IST 027113, eTRIMS - eTraining for Interpreting Images of Man-Made Scenes, and ICT-214975, Co-FRIEND : Cognitive & Flexible learning system operating Robust Interpretation of ExteNded real scenes by multi-sensor Datafusion

# Contents

# Chapter 1

# Introduction

Humans are capable of understanding very complex scenes quickly and with minimal apparent effort. Figure 1.1 illustrates this for a typical everyday scene. Even a quick glance will suffice to identify the main entities (people, trees, road, truck, bicycle), to understand the activities taking place (garbage collection and mail delivery), and to establish the overall context (morning in a suburb). Our interpretation of the scene involves not only the visual information (two people next to a truck), but also inferred information based on domain knowledge (occupation of the people in the scene) and temporal context (activities they are performing). Having observed the scene, we can describe it to another person in a highly compressed form, using shared concepts describing common knowledge about the world. The goal of computer-based scene interpretation is to create an artificial system capable of understanding real-world scenes in order to perform tasks which currently require human supervision, such as evaluating medical images, monitoring airport activities, or content-based image retrieval and annotation. Such tasks have been referred to as *scene interpretation*, *scene understanding*, *image understanding* and *cognitive vision*. In this thesis, these terms are used as synonyms.

The research on scene interpretation tends to deal with one of two main problems. One is the extraction of useful information from images, or *low-level vision*. The other one is the modelling of knowledge about the scene in terms of relations between real world objects and deriving new knowledge through logical or probabilistic reasoning, often called *high-level vision*. Both have been the subject of extensive research for decades and have been successfully applied to a number of problems, such as object detection and tracking, face detection, image indexing and stitching, or modelling complex real-world problems through constraint networks, probabilistic models or grammars.

A complete system capable of understanding natural scenes needs both ingredients, but there is comparatively little research dealing explicitly with the interface between the two layers. It seems intuitive that knowledge about the scene should help to detect objects more reliably, and that reliable detections will in turn help to understand the

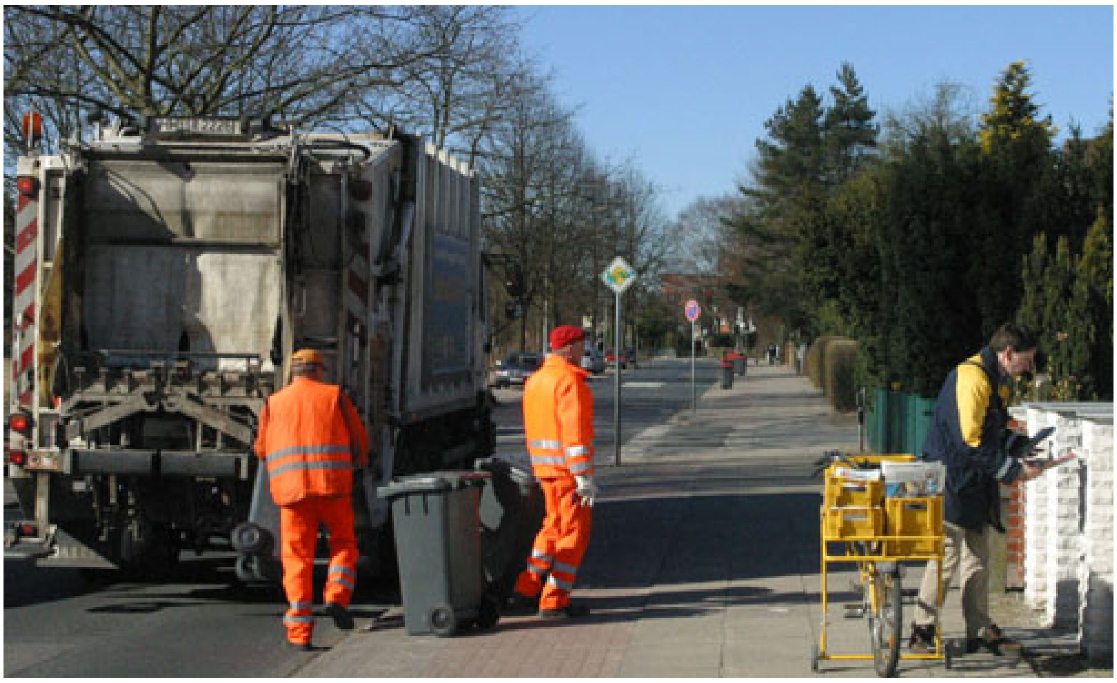


FIGURE 1.1: An image of a natural scene showing garbage collection and mail delivery.

scene, but this has proven to be difficult in practice. A number of scene interpretation systems have been designed in the past, which have tackled this problem in a number of ways, but they still remain confined to narrow domains due to the difficulty of the task, more than thirty years after the first interpretation systems were proposed. The gap between low-level vision and high-level vision is seen as a major reason for this, and is the central topic of this thesis.

## 1.1 Challenges of Image Understanding

The first problem scene interpretation systems face is the visual richness of the natural world. The number of different visual categories recognised by humans is very large, and is widely accepted to include at least three thousand categories [Zhu and Mumford, 2006]. Even the most detailed categorisation datasets (such as Caltech 256 [Griffin et al., 2007]) consist of only a few hundred categories, and many object detection methods are tested on much smaller datasets. Automatic recognition of thousands of categories raises the question of learning visual descriptions of categories automatically from weakly labelled data. Modelling appearance by hand becomes prohibitive and makes it difficult to adapt the vision system to new application domains.

The richness of the visual world also leads to a second problem, which is ambiguity. Due to the large number of categories and a large intra-category variation, the detection of objects is an uncertain process dependent on context. Another source of ambiguity is that the semantic label we use to name an object is not only based on the object's appearance, but also other considerations such as function, leading to the theory of affordances [Gibson, 1977]. Such information is not typically present in images, so high-level knowledge is needed to accurately identify some of the objects in the scene. As a quick example, a billiard cue ball and a white bowling ball might have exactly the same appearance, but they differ is size, weight, material, and how they are used. Such information is not necessarily directly visible in an image, but can be inferred from other visual information in the scene. It is known that the human visual system invokes top-down spatial context at a very early stage of visual recognition [Bar et al., 2006], which enables us to interpret heavily blurred or noisy images. The contextual, top-down component has been seen as important since the early days of image understanding [Ballard et al., 1977], but much of the work on object detection and classification has not made use of scene context.

The third problem is the complexity of vision. It is accepted that a bottom-up combinatorial exploration of image segments is not computationally tractable (see the discussion in [Tsotsos, 1990]) and that a top-down directed search is needed. This has motivated the development of model-based approaches to object detection, which look for specific patterns. Looking for a specific type of object (such as a license plate) is a relatively easy task. Understanding a complex scene filled with objects belonging to hundreds of potential categories is more difficult. On the one hand, knowledge about the scene can help detect and classify objects in an image. On the other hand, scene context can not be established until some reliable observations are obtained. The human visual system solves the problem by activating many parallel visual routines at different levels of abstraction, which work together towards a complete understanding of the scene [Ullman, 1984]. Many researchers agree that an intermediate object-level layer can reduce such complexity and act as a buffer between low-level and high-level processes in a scene interpretation system, but reliable detection of objects at this level is still a complex task due to the need to use both low-level and high-level sources of information.

The three problems outlined in this section are all related to one of the central problems of scene interpretation systems today: accurate detection and classification of objects in natural scenes. Although both the low-level and high-level vision communities have produced complex and powerful systems in recent years, there remains a *semantic gap* between the image level and the symbolic level at which reasoning and understanding take place. This problem has been referred to as *symbol grounding* [Harnad, 1990] in the more general context of Artificial Intelligence and observed in other fields, such as speech recognition. While interpretation systems have bridged this gap for certain applications and object types, the problem of combining bottom-up observations with high-level expectations in a generic way remains a challenge for automatic understanding of complex scenes.

Scene interpretation attempts to explain visual input (low-level information) in terms of accumulated knowledge of the world (high-level models). This implies both bottom-up (observation) and top-down (context) information flow. There seems to be no consensus among vision researchers as to the precise roles of the two, and there are purely bottom-up systems, as well as systems which invoke top-down hypotheses early in the interpretation process. One practical aspect of the integration of low-level and high-level information are the different representations involved. At the image level, the information has the form of a map, with values distributed on a pixel grid. At the understanding level, the scene is typically modelled as object instances and relationships between them. As will be described in Chapter 2, past scene interpretation systems have used a variety of approaches, from using map-based representations exclusively, to modelling small perceptual primitives such as corners and edges using an object-oriented representation.

## 1.2 Goal and Problem Statement

This thesis addresses the problem of generic middle-layer vision. Based on the short discussion of challenges in this chapter and a more detailed review of related work in the next chapter, there are several requirements which a middle layer should fulfill.

1. The architecture should be **generic**, and not dependent on a specific interpretation methodology or low-level segmentation algorithm. This not only avoids the pitfalls of some early scene interpretation systems which were difficult to apply to new domains, but also provides access to a wealth of state of the art algorithms. There are a number of competing approaches in both low-level image analysis and high-level reasoning camps, and a unified platform can provide a way to combine and test them.

2. The architecture should provide an **intermediate representation** of the scene which allows the low-level appearance modeling and high-level scene modeling to be decoupled from each other. This simplifies the creation of domain-specific knowledge bases.

3. Since it is known that object detection is error-prone, the architecture should **model the uncertainty** of object detection and classification.

4. To cope with the ambiguity of visual information, it should be possible to use **high-level context** for improving the detection and classification of objects in the scene.

5. The middle-layer should be able to deal with **incremental interpretation**. Since low-level and high-level processes both depend on each other, an incremental propagation of context can reduce the complexity of finding a solution. Incremental interpretation is also important if the middle-layer approach is extended to dynamic scenes.

6. A standard set of **middle-layer services** which are needed in a typical interpretation system should be defined, such as providing evidence in a bottom-up fashion or confirming high-level hypotheses.

7. Since there are many possible interpretations, the interpretation process may lead to **conflicts**. Certain hypothesised objects might be missing due to **occlusion**, or the high-level model of the scene can conflict with the observed evidence. The middle-layer should provide mechanisms for resolving such issues, by detecting possible occlusions and conflicts and passing the relevant information to the reasoning system.

This work focuses on interpreting static images. A number of other considerations play a role when interpreting time-varying scenes, and a number of additional sources of information could potentially be used, based on the application domain and research context. The extension to dynamic scenes involves tasks such as tracking and modelling movement, which are not addressed in this thesis. Similarly, since the information comes from one modality, the problem of multi-sensor fusion (which is common in autonomous robots) is not adressed.

## 1.3 Short Summary of Results

This thesis presents a generic probabilistic architecture designed to address the issues listed in the last section. It uses an intermediate representation of the scene in terms of object views. The views are manifestations of objects in an image, and are characterised by a set of probabilities relating each instance to different concepts of the high-level ontology. The middle layer knows how to create views from low-level algorithms and pass them to the high-level as detections, with probabilities which model the detection, localisation, and classification uncertainty of the low-level algorithms.

In the other direction, the middle layer can use high-level context to improve the detection and classification performance. High-level predictions can be passed down to the middle layer as view hypotheses to be confirmed, or as contextual priors which can improve classification using a novel contextual classification method. Additionally, a method for passing additional constraints to the low level is presented, which can improve segmentation in a generic fashion.

The main scientific contributions of this thesis are:

- A probabilistic intermediate representation of the image which can be obtained from a number of state of the art object detection algorithms.
- A model for integrating contextual probabilities with appearance-based class probabilities.

- A unified model of detection, classification, and localisation uncertainty which allows for combining multiple detectors and replaces ad-hoc tolerance values.
- A probabilistic appearance model based on decision trees, which is capable of passing expectations to a low-level segmenter.

Unlike many previous approaches, this thesis does not present a solution based on a particular image processing paradigm (such as chaining edges, Gestalt laws or particular types of image patches). Instead, it provides a generic mechanism which works with any object detection mechanism, as long as it provides accurate class probabilities, which is common with many current object detection and categorisation approaches. The integration is demonstrated with a state of the art hierarchical patch-based approach and two region-based approaches, and can be easily applied to other segmentation algorithms.

The architecture presented in this thesis was implemented and used for interpreting images with a number of high-level reasoning systems and low-level detection algorithms. The main experimental results are:

- Contextual priors were shown to improve classification of ambiguous objects in the challenging façade domain, demonstrating the effectiveness of the contextual classification.
- Automatic verification of high-level object hypotheses was used to combine low-level visual structure with high-level compositional structure in order to improve the detection of windows.
- A hierarchical Bayesian Network was used to interpret and segment a façade image by combining the result of several low-level algorithms, demonstrating the ability of the middle layer to effictively abstract away low-level details from the high-level scene model.

## 1.4 Preview of Thesis Work

This thesis is structured as follows. Chapter 2 presents related work in the field of image understanding. It focuses on semantic gap and middle layer architectures, but a short discussion of important interpretation systems is given to provide the needed context. Chapter 3 introduces a new middle-layer architecture based on the goals defined in Section 1.2. Chapter 4 describes the middle-layer software component “Matchbox”, which implements the architecture proposed in this thesis, and addresses the practical issues involved in building such a module. Chapter 5 describes the experimental setup and an empirical evaluation of the presented system. Finally, Chapter 6 provides a discussion of the presented work and an outlook to possible future work.

# Chapter 2

# Related Work

Work on scene interpretation touches on almost every major branch of Computer Science. Signal Processing algorithms are used for examining images, Machine Learning for creating object models, Artificial Intelligence for semantic modelling, Software Engineering for creating a working system, etc. This makes writing a concise overview of related work a challenging task.

The central topic of this thesis is a generic middle-layer platform which can be used as a part of a complete image interpretation system and can tie into state of the art high-level and low-level algorithms in order to provide a complete interpretation system. As a result, this chapter will concentrate on bridging the gap between the image processing layer and high-level knowledge. The insights useful for solving this task are found not only in Computer Vision, but also Robotics, Biology and Artificial Intelligence, which have all tackled the problem with a number of different concepts and architectures.

This chapter introduces the related work in three sections. First, an introduction to scene interpretation is given, including the basic building blocks of an interpretation system and the overview of early work in the field. Then, the approaches for bridging the gap between the image level and the reasoning layer are presented, since they present the most relevant comparison to this work. Finally, the current approaches in object detection and scene categorisation are presented in order to gain some insights into state of the art algorithms.

## 2.1 Scene Interpretation

Humans have the ability to understand complex scenes from a quick glance. It is known that the human visual system is capable of processing visual information at a great speed and that a large number of parallel visual routines cooperate at different levels of abstraction in order to understand a scene [Ullman, 1984]. Marr [1983] offered an influential computational model of human vision, separated into four stages: image-based, surface-based, object-based and category-based.

Any artificial system attempting to achieve the same level of understanding as a human must deal with a number of issues. At the lowest level, it needs to accept the input in terms of a digitised image, and derive an intermediate representation based on salient and stable features, abstracting from the very noisy pixel representation. At an intermediate level, it then needs to group and match such features to concepts from an ontology which defines the types of objects the system can understand. Finally, at the high level, the observed instances of these objects need to be interpreted: what does this particular configuration of objects tell us about the scene, what caused it, and what is likely to happen next? This reasoning step is not necessarily limited to observable values such as spatial relations, but can include higher-order knowledge about laws of physics, human behaviour, etc.

The rest of this section describes the issues involved with machine interpretation of natural scenes and presents an overview of common approaches to both low-level and high-level vision for scene interpretation. Despite decades of intensive research into both fields, reliable detection of objects in complex natural scenes remains an open problem.

### 2.1.1 Semantic Gap

Computer Vision essentially provides a matching of image signals to symbolic representations. However, this problem is not restricted to Computer Vision. It occurs every time when quantitative measurements from the real world have to be matched to a symbolic representation, such as in speech understanding and robotics. For example, early AI systems for understanding sonograms have faced the problem of dealing with noisy signal input [Nii et al., 1982]. In the literature, the problem has been referred to as "semantic gap", "symbol grounding problem", "perceptual grounding" and "signal-symbol mapping". In robotics, where object tracking and manipulation play a large role, it is often referred to as "anchoring".

In machine learning, many algorithms attempt to learn classifiers capable of matching quantitative measurements to class labels. Classification is an important part of the signal-symbol correspondence problem, but the ambiguity present in real-life scenes often means that global and local scene context is needed to improve the detection and classification of object instances in complex scenes. Another important problem is assigning a role to scene objects. Assuming that objects are accurately detected in an image, they still need to be matched to a high-level scene model. There can be a number of valid models explaining the observation, and the same object may play a different role within the same model.

The *connectionist* school of Artificial Intelligence avoids the problem by modelling knowledge in terms of layered neural networks [Fahlman and Hinton, 1987]. A number of connectionist approaches to Computer Vision have been proposed [Fukushima, 2003, Huu et al., 2005], but the majority of scene understanding systems (and many reasoning systems from classical AI) use a symbolic representation for high-level models.

An influential discussion of the Symbol Grounding problem was given by Harnad [1990]. Harnad argues for a hybrid model which reasons above the level of symbols, but uses connectionist methods for grounding the symbols to sensory input. Some Computer Vision researchers have followed this path and used neural networks for finding object instances in images [Rowley et al., 1996]. Another approach are the Conceptual Spaces of Gärdenfors [2000], which introduce a formalism for geometric treatment of knowledge and provide a distance measure for comparing similar concepts. Gärdenfors divides the scene information into three levels: the *subsymbolic level* which directly describes the sensor inputs, the *linguistic level* which describes the scene using a symbolic language, and the intermediate *conceptual level* where objects are described in a metric space using domain-independent cognitive dimensions. The conceptual spaces are seen as a middle way between connectionist and symbolic approaches. They have been extended since their introduction [Rickard et al., 2007] and applied to the anchoring problem in robotics [Chella et al., 2004].

A logic-based treatment of the problem was given by Shanahan [2005], who sees the interpretation of noisy sensor inputs as a form of logical abduction. Abduction is more commonly used to reason at a higher level, but in this work, it is applied to very low-level input, such as the output of a Sobel operator. A set of sensory inputs can generate a number of object hypotheses which make predictions about further observations and are then tested in turn. The testing may involve action, such as a robot nudging or moving the object to provide further information.

Probabilistic approaches to scene modelling could also model the low-level phenomena using large JPDs and thus avoid the semantic gap in a manner similar to the way connectionist models do, but the complexity of such probabilistic models has pushed the probabilistic models towards reasoning with symbolic object representations [Rimey, 1993, Neumann, 2008]. Such symbolic representations still need to be accurately matched to low-level observations. A survey of approaches for dealing with the Symbol Grounding Problem by Taddeo and Floridi [2005] concluded that the problem is important, but unsolved.

This thesis concentrates on the semantic gap in image understanding systems and the issues specific to Computer Vision. For example, it is known that the three-dimensional nature of natural scenes makes the construction of grammatical models more complex than with language processing, which is one-dimensional [Zhu and Mumford, 2006]. The huge number of categories in the visual world and the large intra-category variation remain difficult obstacles for the development of reliable object detectors. It should be pointed out that many scene interpretation systems do not deal with pure symbols, but typically use a combination of symbolic knowledge (presence of an object) and quantitative information (location and dimensions of the object). This chapter introduces the architectures proposed for dealing with this problem, both in the field of categorisation and object detection, and when applied to the larger context of scene understanding.

### 2.1.2 Early Scene Interpretation Systems

Research into interpreting static images goes back to the mid 1960s and the grammatical image parsing of Kirsch [1964] and Narasimhan [1966]. The work of Clowes [1971] and Fu [1982] extended the grammatical formalism to define relations between scene objects and reason about simple artificial scenes. Although the low-level vision routines at the time were not able to detect scene objects reliably, the hierarchical high-level modelling of objects in the scene has influenced much of later research. A similar approach was later used by Ohta [1985] to model natural outdoor scenes.

Early image understanding researchers were already aware of the influence of top-down control in image analysis. The vision system of Ballard et al. [1977], was attention-driven and made use of context and high-level hypotheses. The scene was modelled in terms of concepts subject to high-level constraints, and the interpretation of the image (the so-called "sketchmap") was the instantiation of the model concepts which were observed in the image. The possible interpretation of low-level detections was described by "mapping procedures", and governed by a set of "executive procedures" which select the best mapping of the low-level edges to the modelled concepts. The mapping procedures, which describe the *visual* knowledge about the objects, were constructed by hand for domains such as locating docked ships and finding ribs in a radiograph.

Another approach for interpreting natural scenes stems from the aerial image understanding work of Hwang and Matsuyama [Nagao et al., 1979], implemented as the image understanting system SIGMA [Matsuyama and Hwang, 1985, 1990]. SIGMA was capable of bottom-up and top-down processing, by implementing a coarse-to-fine strategy where low-level algorithms detected potential object locations and focussing attention on the relevant sections of the image by starting specialised object detection algorithms trained to detect specific classes of objects. This coarse-to-fine strategy saved processing time and improved image segmentation. Another system capable of interpreting aerial scenes at that time was the SPAM system [McKeown et al., 1985], which was also successfully applied to airport scenes. SPAM used production rules to reason about a scene, making it an expert system, and was capable of segmenting and providing ranked interpretations of aerial images.

The VISIONS system [Hanson and Riseman, 1978] was an early image understanding system based on a blackboard architecture, designed for interpreting colour images of natural scenes. Later, the VISIONS system was extended by the notion of *schemas*, which were active processes that encapsulated knowledge about a specific object class, thus turning VISIONS/Schema into an active agent system [Draper et al., 1988]. The system was capable of exploiting scene context for improved object detection, such as triggering the detection of telephone poles when a road was detected.

The PSEIKI system [Andress and Kak, 1988] was another blackboard-based system which could interpret images by matching the detected edges to the knowledge represented as a line drawing of the scene. The drawings could be automatically generated for some domains, like extracting roads from cartographic information in order to aid airplane localisation.

The earliest system capable of understanding scenes involving motion known to the author is [Badler, 1975]. It proposed a methodology for conceptual modelling of scene objects and movements using rules, with a hierarchical representation for increasingly complex movements. The system was capable of giving a natural-language description of the scene.

Early work on video understanding also includes the work in Hamburg [Nagel, 1977]. The group produced the early interpretation system for traffic analysis system called NAOS [Neumann and Novak, 1986]. Like Badler's early work, this system could also use knowledge-based techniques to analyse traffic scenes and describe them using natural language. The scene was described by a static model of the environment and a dynamic description of object positions (cars) at different time points. Based on these data, simple and complex events could be detected, such as turning, stopping or overtaking. Such intuitive qualitative concepts were used to provide a natural language description of the scene. Since these concepts were defined in a declarative manner, the high-level scene model was both compact and understandable. The work on traffic scenes was continued by Nagel [1988] using *Situation Graph Trees* for activity modelling. Both systems have been continually developed and the concepts are still used for interpreting new domains today [Hotz and Neumann, 2005, Terzić et al., 2007, Fernández Tena et al., 2007].

Although most of the early systems were ambitious, they suffered from the lack of reliable object detections. They performed well within the narrow confines of their application domains, which featured few objects and tight constraints between them. As argued in [Draper et al., 1996], the development towards less constrained scenes and many objects belonging to many categories has rendered some of the design decisions of these early systems obsolete.

On the other hand, the field of Computer Vision has improved drastically since the early days of scene interpretation systems. Many techniques were developed for realiable

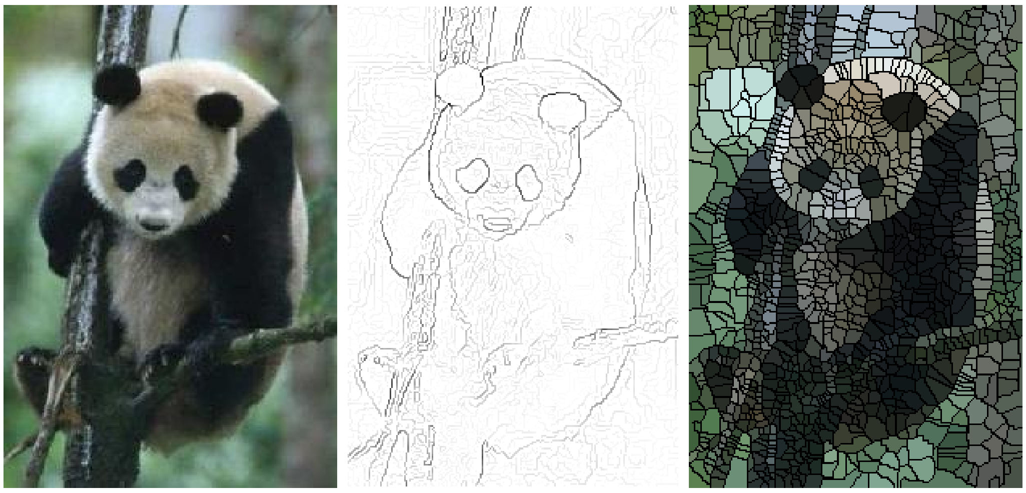


FIGURE 2.1: Classic image processing tools. Left: original image. Middle: edges detected by a Canny detector. Right: regions detected by a watershed transform. The results can be recognised by a human, but neither approach detects the complete object of interest – in this case a panda.

detection of certain object classes, and the increase in computing power made many new approaches possible. Although reliable object detection is far from being a solved problem, today's scene interpretation systems can draw on a much more advanced set of low-level algorithms.

### 2.1.3 Low-Level Image Processing in Interpretation Systems

There are two major approaches to detecting objects in images used today. The first one is based on detecting boundaries between regions in the image and attempting to model objects in terms of their contours. The contours are manifested in images either as edges or corners, both of which are very well understood and can be detected reliably. There is ample evidence that object shape plays a large role in biological object recognition [Palmer, 1999, Chap. 9].

This approach is prevalent in scene interpretation systems, which either segment regions which are assumed to correspond to an entire object, or try to string together edge segments and corners into complete objects. But, as Figure 2.1 shows, natural images contain many regions and edges, most of which do not correspond to objects or object boundaries. The oversegmentation issue has led into research on hierarchical representation of regions, such as the region hierarchies of [Todorovic and Ahuja, 2008, Drauschke, 2009] and map representations like [Meine, 2009], but reliable detection of complete segmented objects requires top-down information. In the case of videos, the temporal component introduces additional knowledge which makes it possible to segment out complete objects automatically, as in [Cavallaro and Ebrahimi, 2004], which simplifies the processing, but often introduces other problems.

Due to these problems, scene interpretation systems have typically been restricted to very specific domains where segmentation is easier, and their performance has suffered due to the inability to accurately model the uncertainty of low-level detections. The next section will discuss some of the examples.

The second approach focuses on modelling appearance of the object directly, which has led to a number of successful frameworks for bottom-up object detection and classification. It is based on finding representative image patches which contain important appearance information and can be reliably detected in images, and then constructing appearance models which combine such patches to describe object appearance. Such image primitives can be biologically motivated, such as textons [Julesz, 1981] and symbolic tokens [Marr, 1983], or designed to be efficiently computed, like Haar-like features [Papageorgiou et al., 1998]. More recently, patches are often detected using an interest point operator such as Harris, Förstner or Difference of Gaussians (DoG) – see Figure 2.2 for a comparison – and then clustered into a codebook of representative parts, so the repertoire of important patches is learned [Weber et al., 2000].

The patches are either described as subsampled images of a certain size (typically 10x10 to 30x30 pixels) and compared using correlation-based methods, as in [Burl et al., 1998, Leibe and Schiele, 2003] or described using a numerical descriptor, as in [Lowe, 2004, Ke and Sukthankar, 2004]. The choice of a particular interest point detector is not too important in practice [Mikolajczyk and Schmid, 2005], but the choice of the feature point representation is, and depends on the application. The representation can be specific to a particular object instance such as SIFT [Lowe, 2004], to a particular object category [Leibe and Schiele, 2003], or be general enough to be shared among many different categories [Krempp et al., 2002, Torralba et al., 2004]. Some approaches build hierarchies of features, from very general to class-specific [Fidler and Leonardis, 2007, Ommer and Buhmann, 2010].

Several recent approaches have combined segmentation with part-based models, by using patches with segmentation masks [Leibe et al., 2005], using small regions as basic building blocks and describing them using a feature descriptor [Todorovic and Ahuja, 2008, Gu et al., 2009], or by explicitly using both patches and regions together in the same probabilistic model [Li et al., 2009].

The success of modern object detection methods for detecting hundreds of classes makes them attractive for the problem of reliable object detection in complex natural scenes, which is an essential requirement for reliable scene interpretation. Some methods relevant to scene interpretation are discussed in more detail in Section 2.3.

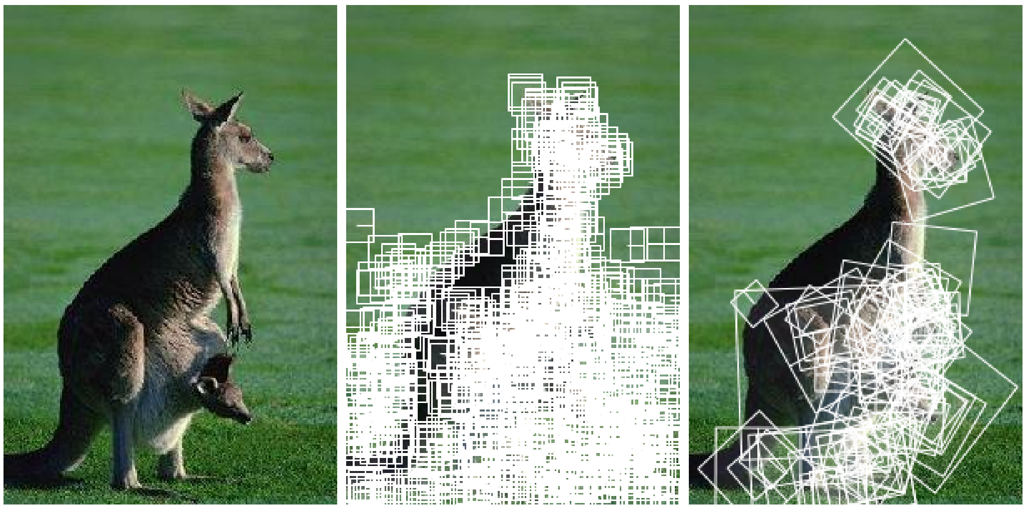


FIGURE 2.2: Results of two different interest point detectors. Left: original image. Middle: image patches around isotropic Harris corner detections at scale 1. Right: image patches around DoG interest point detections (size of the box is proportional to the scale). The individual detections must be combined in order to detect the kangaroo.

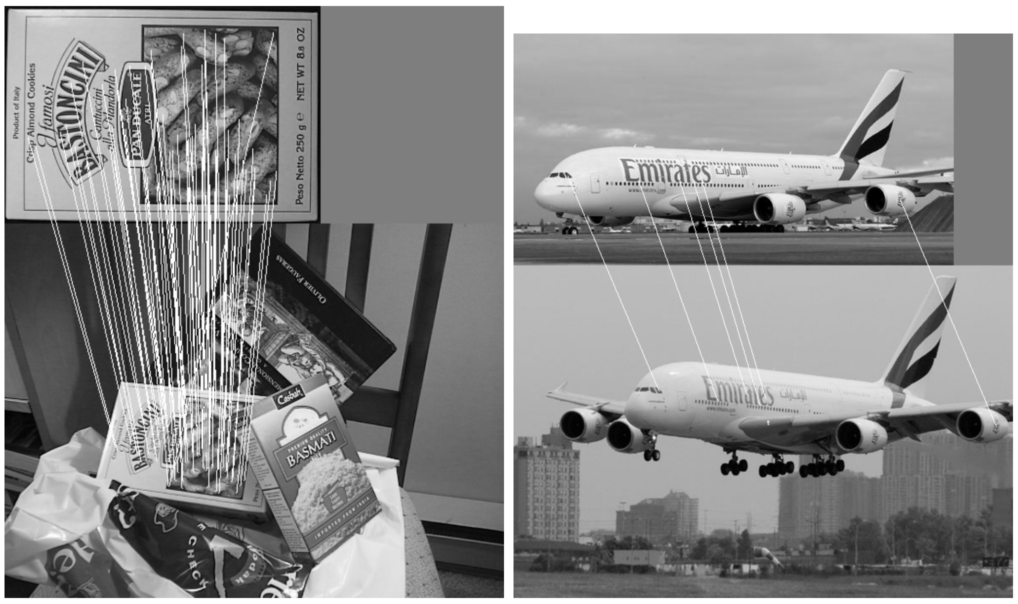


FIGURE 2.3: Matching of SIFT features in two images. Left: an example on a canonical image. The object exemplar is shown at the top, and found in the image below, despite rotation, scaling and perspective effects. Right: two similar airplanes from different scenes. It can be seen that very few matches are found. SIFT features are strong for finding the exact object, but do not generalise to similar objects from the same class.

### 2.1.4 High-Level Scene Interpretation

High-level reasoning has also seen ample development since the time of the earliest interpretation systems. A number of researchers continued to refine the hierarchical logical models of scenes and model the relations between the parts using crisp rules or constraints. The second approach attempts to understand the scene in terms of probabilities and to model the likelihoods of object configurations. In recent years, hybrid models have also gained ground, which use both crisp and probabilistic constraints, such as stochastic grammars.

#### 2.1.4.1 Crisp Scene Models

Crisp interpretation systems use crisp rules to reason about observed objects. This often not only involves deductive reasoning, but also model building or abduction. The domain is modelled in a knowledge base and described by an ontology and a set of rules which describe valid instances of higher-order concepts. One way to understand scene interpretation is as abduction, as argued by Shanahan [1989]. While deductive logic can be used to make predictions based on known rules, abduction can provide an explanation of the scene. Abduction was also applied to scene interpretation by Cohn et al. [2002].

Logical interpretations typically use knowledge representation paradigms from Artificial Intelligence, such as Description Logics [Neumann and Möller, 2008] to represent the knowledge about the domain. DL engines such as RACER have been used for interpreting natural scenes [Hummel et al., 2008], but other systems can be shown to provide equivalent functionality, such as the configuration tool KONWERK, used in the SCENIC interpretation system [Hotz and Neumann, 2005, Terzić et al., 2007, Hotz et al., 2008, Terzić et al., 2010]. KONWERK approaches scene interpretation as partial model construction and uses logical inference, part-whole reasoning and constraint propagation to arrive at an interpretation of the scene, which is represented as a partonomical tree consisting of a scene decomposed into its parts. Each part of a scene is seen either as a *primitive* (an object which cannot be decomposed further) or an *aggregate* consisting of several parts, which are governed by aggregate-specific constraints. Figure 2.4 shows the ontology used for interpreting images from the façade domain. A similar approach models the scene using the JESS rule engine [Bohlken and Neumann, 2009].

Grammars also have a long history in the field of scene interpretation. One of the first attempts to extend the grammatical formulation of images beyond the object level and to model complete scenes was the work of Clowes [1971] on interpreting line diagrams and simple scenes. Another early application of grammar for scene understanding was the early work of Fu [1982], who introduced *syntactic pattern recognition* to model the relations between the objects in the scene. The scene (the famous Roberts' "blocks

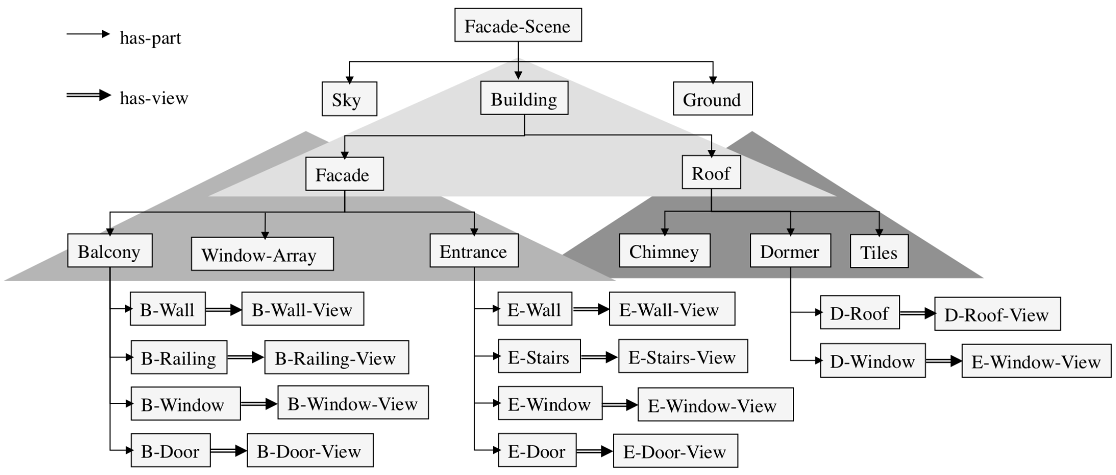


FIGURE 2.4: Ontology used to interpret façade scenes with the SCENIC interpretation system. The aggregates are represented as triangles. The *Views* at the bottom represent an interface to the low-level detections, and are described in Chapter 3.

world") was modelled as a compositional hierarchy of objects, which could be decomposed into faces. This work struggled with the lack of reliable detections of objects. The modelling of shape using grammars has also been a research topic, building on the *medial axes* proposed by Blum [1973]. For example, Leyton [1988] used formal grammars to model shape deformation. These *process grammars* were able to infer processes which deformed the shape of natural shapes such as tumors. Another approach represents shapes as *shock graphs*, like in the work of Kimia et al. [1994].

More recent work on grammar-based scene interpretation includes the facade interpretation of Čech and Šára [2007] and [Ripperda and Brenner, 2009], or scenes involving parking lots [Seo et al., 2009]. While the examples given do show the similarity to other crisp reasoning systems such as description logics and compositional hierarchies, this section can not provide a complete overview of grammar-based work on image understanding. A more complete overview can be found in [Zhu and Mumford, 2006].

Much of the work on scene modelling has dealt with dynamic scenes, such as interpreting actions and behaviour. In this case, the primitives are often object trajectories instead of object detections. The work of Nagel [2004] models complex behaviour as combinations of simpler *occurrence* concepts. The ocurrences are described by a set of fuzzy predicates which have to hold in order to match a tracked trajectory to one of these concepts. Such *movement primitives* are combined into *Situation Graphs*, which model admissible sequences of occurrences. Each node of such a graph represents a state (defined by a set of predicates), and the edges represent transitions between the states, allowing for prediction of following events. Each node can be refined, by linking it to another Situation Graph which models the state in more detail, giving a hierarchical structure called a *Situation Graph Tree*. This approach has been used to model a number of domains, from traffic scenes to human behaviour [Fernández Tena et al., 2007]. A similar hierarchical approach for modelling activities was used by Kojima et al. [2002]

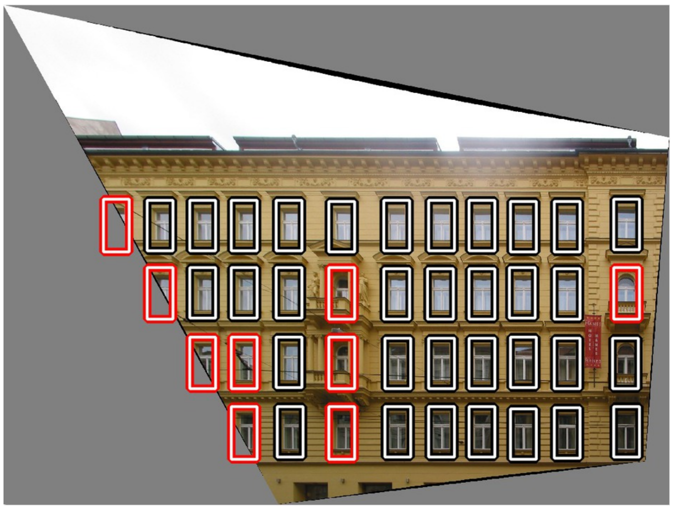


FIGURE 2.5: An example of high-level hypotheses. The white detections are the result of a low-level window detector. The red rectangles are hypotheses of further windows, created by the SCENIC system based on the façade structure. The hypotheses complete the understanding of the scene in the cases where the low-level algorithm fails to detect objects due to shadows or occlusion, and can be used as a source of context.

for interpreting human behaviour. An activity can be specialised into more specific actions by adding additional predicates which describe the action more precisely.

Another interpretation track focuses around the work of Thonnat et al. One recent example is the AVITRACK system for monitoring airport scenes [Fusier et al., 2007]. A scene is described in terms of static and dynamic spatial zones which are associated with particular activities. The activities are modelled in terms of primitive and composite *states* and *events* and described using a novel language designed to express expert knowledge in a declarative way. The system can interpret complex aircraft preparation videos by recognising the modelled activities in real time.

Further logic-based approaches include Prolog rules [Shet et al., 2005], Inductive Logic Programming [Muggleton and de Raedt, 1994] and fuzzy temporal logic [Gerber et al., 2002, Nagel, 2004].

What all crisp interpretation systems have in common is the representation of the scene as instances of objects from an ontology, restricted by crisp rules and relations allowing for logical reasoning. They expect classified concept instances as input, and can provide hypotheses of missing and inferred objects and events as explanation and potential feedback information. While this type of modelling is very useful for many problems, the essential drawback is the lack of a quality measure of an interpretation. Crisp reasoning systems can create a number of possible explanations of the scene, but unlike probabilistic approaches, they have no way of picking the "best" one.

#### 2.1.4.2 Probabilistic Scene Models

One way to reason about a scene is to represent the relations between the scene objects as a large joint probability distribution. Representing complex scenes as large probability distributions quickly runs into complexity problems, which is why Bayesian Networks are generally used to exploit the independence of some of the variables, like in the work of Binford [1988]. Further examples of using Bayesian Network technology for scene interpretation include Causal Probabilistic Networks [Jensen et al., 1992] and Bayesian Belief Networks [Buxton and Gong, 1995]. The first attempt to bring crisp structure to Bayesian Networks was the work of Rimey and Brown [1994], which used a crisp compositional hierarchy to model the physical structure of a scene (a tabletop scenario). Given the location of a dish, the location of other objects in the scene could be determined, and active vision tasks executed. More complex object-oriented representation of structured objects was explored by Koller and Pfeffer [1997] and Gyftodimos and Flach [2002]. Despite these improvements, the complexity still remained a problem for modelling complex scenes, leading researchers to combine Bayesian Networks with crisp partonomies, as described in the following section.

Markov Random Fields have traditionally been used for low-level segmentation tasks, but have recently also been applied to high-level vision and modelling constraints between objects. The optimal labelling of image regions is defined as the maximum a posteriori (MAP) probability estimate of the truth, the best that one can obtain from random observations [Li, 1994]. The relations between scene objects are defined locally, between nearby objects, allowing for parallel processing. For example, Modestino and Zhang [1992] define an MRF over image regions by specifying the clique functions for single and pair-wise cliques. The weights used to define the potential functions are set manually. Carbonetto et al. [2004] automatically learn the potential functions, but treat the regions as a *bag of words*, not making use of spatial relations between objects, only co-occurrence. Heesch and Petrou [2009] propose an asymmetric MRF where spatial and topological relations are modelled between regions in small neighbourhoods. The labels are initialised randomly and the final labelling is obtained through probabilistic relaxation of the field. Each region is represented by a set of labels, each associated with the probability.

Some authors have extended the region-based classification using Markov Random Fields down to the pixel or superpixel level. In this sense, these architectures provide a context-aided segmentation. The work of Korč and Förstner [2008] models the super-pixel neighbourhoods for eight classes from the façade domain and uses it to arrive at a labelling of the image. The work of Shotton et al. [2009] defines objects using a boosting techniques on image primitives called *textons* [Julesz, 1981] and defines a CRF over the image regions, leading to a segmentation and labelling of the complete image. Such approaches clearly reach down to the middle-level of image understanding, as they operate on regions and texture, but their labelling power comes from the ability to model relations between objects in a scene.

In the field of activity modelling, a number of researchers have applied Hiden Markov Models to learn and represent activities. For example Brand [1998] represents behavious in an office environment by an HMM and uses entropic priors for fast learning of concice models. Galata et al. [2002] use variable-length Markov models for modelling vehicle behaviour in traffic scenes. The work of Hongeng et al. [2000] introduces a hierarchical HMM representation which can be used to model complex activities involving many actors.

The advantage of probabilistic models is that they can provide a probabilistic evaluation of the result. This can be a useful way to provide context to lower levels of the interpretation system, for example, by using the classification probabilities provided by a Markov Random Field.

#### 2.1.4.3 Hybrid Models

A number of hybrid approaches to combine logical and probabilistic reasoning for scene interpretation have arisen in recent years. Some of them can be found in [Cohn et al., 2008]. Two approaches are briefly introduced in this section.

The attempt to use structured Bayesian Networks for scene interpretation led to the design of Bayesian Compositional Hierarchies (BCH) [Neumann, 2008]. The BCH models a scene as a crisp partonomical hierarchy, starting with the scene as the root node and ending with primitive (atomic) objects at the leaves. Each structured object (or *aggregate*) contains the full JPD representing the relations between the attributes of its parts, but not the relations to parts of other aggregates (see Figure 2.6). A propagation scheme is introduced, by which the context introduced by fixing attributes of one object in the tree can be propagated upwards and downwards through the hierarchy and affect other objects. The strength of the BCH is that its compositional structure mirrors the partonomical structure of logical systems such as Description Logics or grammars, so they can be used to guide the logical interpretation process by suggesting which decisions are more probable based on the observed objects. At the same time, as shown in [Kreutzmann et al., 2009], the BCH can also be used as a source of prior probabilities for a low-level classifier. The application of the BCH for interpreting man-made scenes and aiding low-level segmentation was described in more detail in [Neumann and Terzić, 2010].

One other major avenue of development of hybrid systems have been probabilistic grammars, which build on a long history of grammatical parsing of images (even the early work

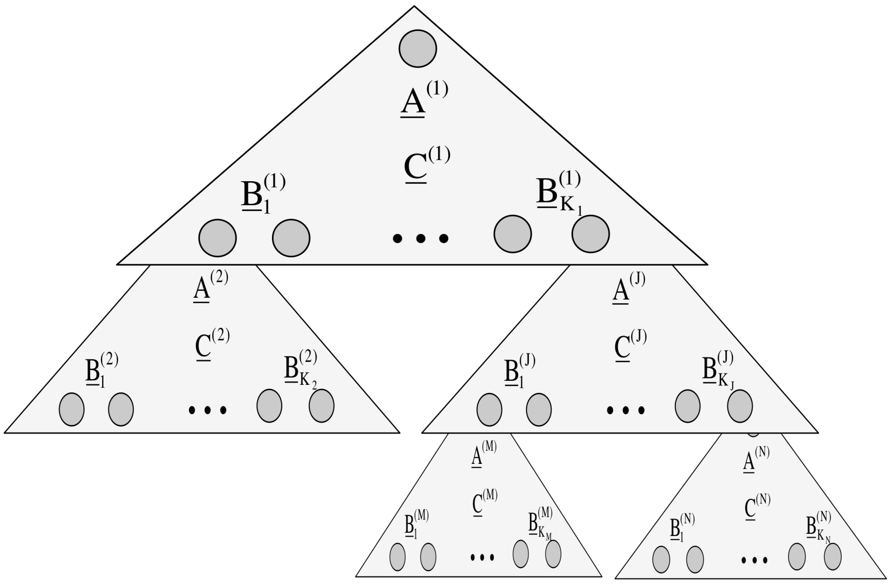


FIGURE 2.6: The structure of a Bayesian Compositional Hierarchy. The triangles denote aggregates consisting of parts. The parts are described by attribute vectors $B_k$, and the entire aggregate by its export attributes $A$. The relationships between the parts are expressed by $C$ (e.g. distances between the parts, or their size). Each aggregate is individually described by the JPD $P(A, B_1 \ldots B_k, C)$.

by Fu [1982] used stochastic grammars). The probabilistic modelling allows such grammars to deal with very noisy information at lower levels, resulting in object and scene models which extend the high-level formalism to the level of the image, thus peforming vision tasks at the middle-layer level. Important examples include the probabilistic grammar models of license plates of Jin and Geman [2006] and the stochastic grammar of images proposed by Zhu and Mumford [2006], introduced later in this chapter. Stochastic grammars were also used to interpret façade scenes by Čech and Šára [2009], where the relations between windows are formulated as a grammar. Such methods can act as a middle-level source of structure, as will be explained in Section 3.5.5.

## 2.2 Middle-Layer Architectures

A number of middle-layer approaches were suggested in the literature for bridging the gap between low-level sensory input and the high-level representation by providing an interface between image processing algorithms and a high-level layer. This section introduces the architectures which were designed to fit into a complete interpretation system and are not dependent on a specific domain.

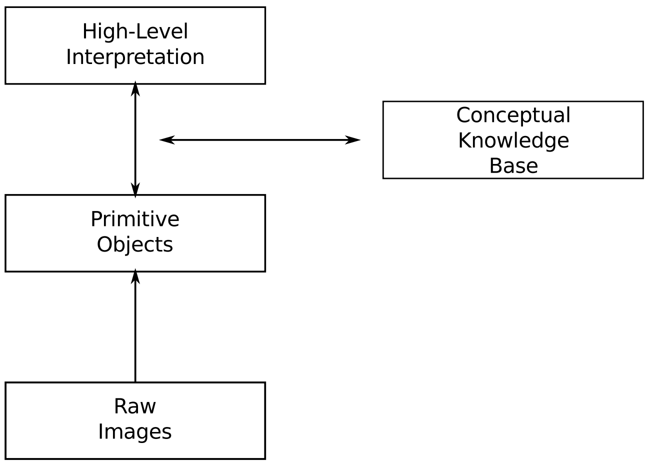


FIGURE 2.7: Basic architecture of a knowledge-based interpretation system. Reproduced from [Neumann, 2002].

### 2.2.1 Bottom-Up Approaches

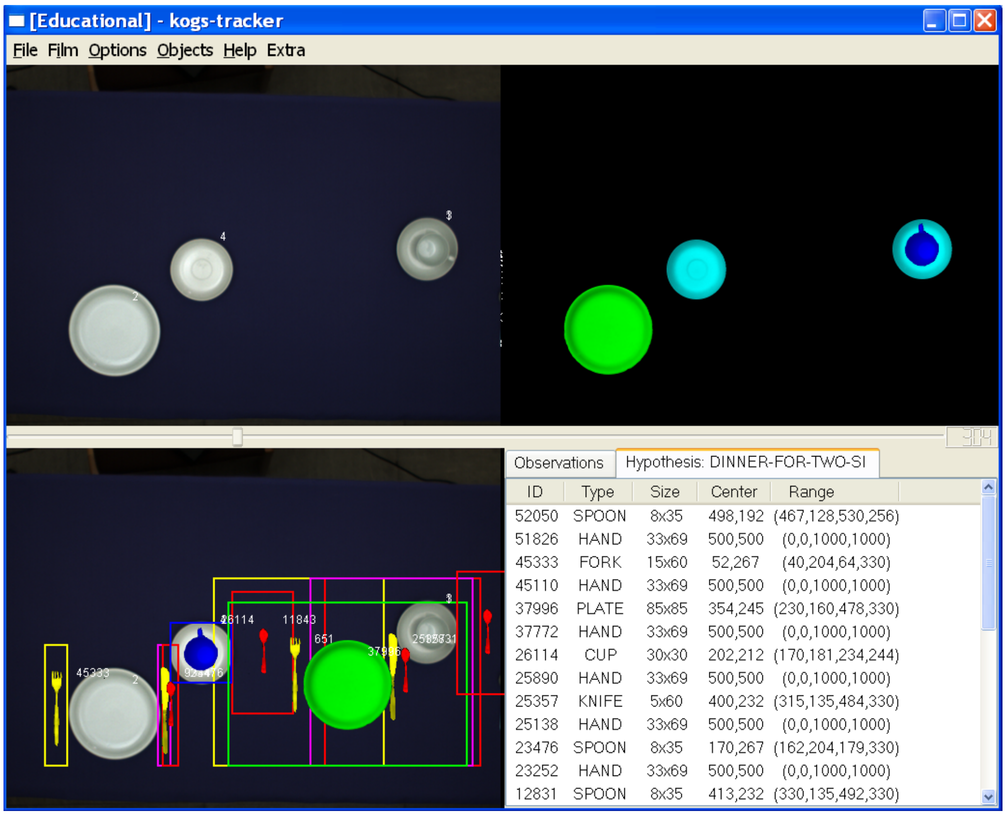
The most common approach to connecting visual detections to a high-level reasoning layer is the simplest one: a strict separation between two independent layers:

- low-level layer, which detects and classifies object instances in a scene, and a
- high-level layer, which accepts the objects and interprets the scene by exploiting the relationships between the observed objects.

Figure 2.7 shows the basic architecture of a knowledge-based interpretation system (taken from [Neumann and Terzić, 2010]). Although the reasoning above the object level may use top-down inference, the low-level detections are accepted as correct and are not affected by the high-level context. The approach is naïve because of the implicit assumption that the low-level and high-level tasks can be completely decoupled from each other and that the scene can be interpreted in a bottom-up fashion, without significant feedback. This simplifies the design of the interpretation system, and avoids the problems of mapping symbolic class expectations to the signal level. The objects are typically described by a symbolic class label and quantitative time and position attributes.

This paradigm can also be seen in Computer Vision literature, where most object detection schemes attempt to solve the detection and classification problems using low-level approaches, and high-level reasoning systems typically work above the level of individual objects.

One of the first architectures for interpreting videos was proposed in the mid 1970s [Nagel, 1977]. The system, faced with the limited hardware power at the time, was based on a low-level layer which extracted a 3D description of the scene using no domain knowledge by extracting the moving objects in the scene. A detailed description


[Educational] - kogs-tracker
File Film Options Objects Help Extra
Observations
Hypothesis: DINNER-FOR-TWO-SI

| ID | Type | Size | Center | Range |
|---|---|---|---|---|
| 52050 | SPOON | 8x35 | 498,192 | (467,128,530,256) |
| 51826 | HAND | 33x69 | 500,500 | (0,0,1000,1000) |
| 45333 | FORK | 15x60 | 52,267 | (40,204,64,330) |
| 45110 | HAND | 33x69 | 500,500 | (0,0,1000,1000) |
| 37996 | PLATE | 85x85 | 354,245 | (230,160,478,330) |
| 37772 | HAND | 33x69 | 500,500 | (0,0,1000,1000) |
| 26114 | CUP | 30x30 | 202,212 | (170,181,234,244) |
| 25890 | HAND | 33x69 | 500,500 | (0,0,1000,1000) |
| 25357 | KNIFE | 5x60 | 400,232 | (315,135,484,330) |
| 25138 | HAND | 33x69 | 500,500 | (0,0,1000,1000) |
| 23476 | SPOON | 8x35 | 170,267 | (162,204,179,330) |
| 23252 | HAND | 33x69 | 500,500 | (0,0,1000,1000) |
| 12831 | SPOON | 8x35 | 413,232 | (330,135,492,330) |

FIGURE 2.8: SCENIC interpretation system applied to the table setting domain. Top-left window shows the video input. The video is processed by a low-level tracker unit, which delivers classified objects, showed on the top-right. The bottom-left window shows the result of the interpretation, together with all the hypothesised objects and their expected positions. The bottom-right window lists the hypothesised objects and their properties.

of the interpretation system is given in a later overview Nagel [2004], based on the experience of the previous few decades. The architecture consists of a *core computer vision* subsystem which delivers a geometric description of the scene, called the Conceptual Primitive Layer. A *conceptual representation* subsystem accepts this input and reasons about the observed scene. The low level system can perform complex 3D tracking by matching 3D object models to the observed data, introducing a top-down component which considerably improves the tracking performance, but does not use scene context above the object level to aid the low-level algorithms.

NAOS [Neumann and Novak, 1986] was an early system for interpreting complex scenes and describing them using natural language, with the application domain of analysing traffic scenes, which added the ability to use top-down interpretation steps above the level of detected objects. The system uses an intermediate description of the scene, called Geometric Scene Description (GSD), which contains information about which object is located where at which point in time. This description is enriched by a symbolic description of object's visual properties (such as colour), so the original image sequence can be generated from the GSD, at least in an approximate form. The reasoning layer takes the GSD as input, assuming that it was generated by a lower-level process, and that it is an accurate description of the real-word scene captured by the camera images. The authors point out that computer vision algorithms of the time were not capable of providing such a description for complex scenes, despite years of concentrated effort in that direction. Unfortunately, this statement is still just as valid today.

The GSD was used in later experiments [Terzić et al., 2007], such as the modelling of table laying scenarios (see Figure 2.8), where a low-level tracker unit segmented moving blobs from a video, classified them using a shape-based method, and provided the GSD to a high-level reasoning layer. The reasoning layer then interpreted the activity and hypothesised future actions and objects. However, even in such a simplified domain, the low-level detection and classification was not easy, particularly under the presence of shadows and occlusion. As a result, the class of an object could change between neighbouring frames, for example by changing from a saucer to a cup, depending on the amount of occlusion. This ambiguity was resolved by using the taxonomy to find the most specific generalisation of the concepts in question (in this example: `saucer` and `cup` are both specialisations of the `dish` concept), and letting the high-level perform the specialisation using the high level models (e.g. cup on a saucer is an allowed configuration, a saucer on top of a saucer is not).

There are two reasons why this way of dealing with classification errors is not optimal. First of all, there is no guarantee that concepts which are close to each other in the taxonomical hierarchy are also similar in appearance. This means that the most specific generalisation will often be the root node of the taxonomy, essentially leaving the classification task to the high level. Secondly, the knowledge base should ideally model domain-specific knowledge, and not be concerned with modelling the inaccuracies of a specific low-level detector in an ad-hoc manner.

A more complex architecture for cognitive vision was proposed by Chella et al. [1997]. Built on the concepts proposed by Marr [1983] and Gärdenfors [2000], their system attempts to construct a bottom-up 3D representation of the scene built from primitive 3D shapes, which is then transformed into an object-centered “mental” image of the scene. This representation is similar to the Geometric Scene Description proposed by Neumann. The reasoning proceeds as an interplay between the 3D “mental” image and the symbolic knowledge base, which defines the objects as combinations of primitive 3D shapes. The need for a bottom-up 3D reconstruction of the scene limited the experiments to very simple scenarios.

One common reason for using feed-forward, bottom-up detection and classification of objects is when realtime constraints need to be met. The AVITRACK airport monitoring system [Fusier et al., 2007] was explicitly designed without feedback in mind. The video streams from several cameras enter a low-level Scene Tracking module, which tracks the objects in the video streams and fuses them together in order to provide an

XML description of the scene which is passed to a long-term tracker and subsequently to the Scene Understanding layer. The tracking itself is very complex in order to deal with different weather conditions and enable around the clock operation. A two-stage classification approach is applied. First, the general class of the object is determined based on simple descriptors, e.g. vehicle, person or equipment. A more complex classification task is started in the background, and it fits a 3D model to the object, e.g. specialising a vehicle into a tanker or a loader. While the low-level algorithms do entail top-down and bottom-up steps, and heuristics are used for long-term tracking (thus representing a form of domain knowledge), there is no downward flow of information from the higher layers. In other words, the description of the scene is created using low-level information, and scene context is not used to aid the low level.

Several interpretation systems were described in this section to illustrate feed-forward, bottom-up approaches, but there are many other systems built on the same principles, such as [Kojima et al., 2002]. Such bottom-up approaches to object detection often work well in practice, as long as the domain allows for accurate detection and classification of objects at a lower-level. However, experience from many projects shows that interpretation systems based on correctly identified objects have problems dealing with detection errors, leading to more complex models which deal with detection errors of particular low-level detectors instead of modelling high-level scene knowledge. Due to this, the research on understanding complex, natural scenes and detecting hundreds of object categories has pushed towards using high-level expectations for improved low-level classification on one side, and modelling detection uncertainty for more robust interpretation under uncertain conditions, as is presented in the rest of this chapter.

### 2.2.2 SIGMA

One of the earliest attempts to bridge the gap between low-level detections and high-level semantic concepts dates back to the work on the SIGMA image understanding system for interpreting aerial images [Nagao et al., 1979]. The knowledge-based segmentation in SIGMA is performed by three independent agents:

- **Geometric Reasoning Expert (GRE)**, which uses a symbolic hierarchical model to represent possible spatial relations between semantic objects,
- **Model Selection Expert (MSE)**, which reasons about most promising appearance models to use when searching for objects in the image, and
- **Low Level Vision Expert (LLVE)**, which extracts information from the image based on a model provided by the MSE.

The three sources of knowledge map neatly to the concepts of a high-level resoning system (GRE), low-level detection algorithms (LLVE) and a middle layer which mediates between the two (MSE). In this sense, the MSE is the interface between the scene domain (semantic object categories) and the image domain (low-level detections). The appearance models define how real-world objects (e.g. “house”) are manifested in the scene (e.g. “rectangular house”), thus abstracting this specialised knowledge from the general model of the scene. The appearance models can be simple (such as a rectangle), or more complex, using part-whole relationships to compose more complex objects from primitive parts.

The image interpretation process is started by the MSE, which chooses objects that have simple appearance, since such objects are easier to detect in a bottom-up fashion. It then queries the LLVE to provide detections for such objects and passes detected object instances to the GRE. The GRE uses the spatial relations between the detected objects to create hypotheses about further objects. Such object hypotheses are passed to the MSE, which looks for the objects in the image. It uses the appearance models for the required objects to ask the LLVE to provide low-level primitives which could be produced by these objects.

In some cases, the evidence found by the LLVE can be conflicting. For example, after the MSE attempts to find roads and houses (which are both expected in the scene and can be described using simple geometric models), some of the regions delivered (call them E1 and E2) by the LLVE can belong to either. Since all image processing in SIGMA is done in a top-down manner (i.e. object hypotheses define where to look in the image, even in the initial step), a conflict arises when two hypotheses overlap, and a piece of evidence can be assigned to both. Sigma defines three situations in which the conflict is not a problem:

- the object categories suggested by E1 and E2 are the same (both can be interpreted as houses),
- one of the two categories is a specialisation of the other, or
- one of the two categories can be a part of the other (a house is a part of a house group).

In the case that none of these conditions apply, the two regions are described using an in-conflict-with (ICW) relation, such as expressing that a region cannot be interpreted as both a house and a road at the same time. Such a conflict is resolved in the high-level, by merging conflicting hypotheses (such as road segments trying to explain the same region in the image) or picking one of the two (if a region is claimed by both a road part and a house group).

This type of conflict resolution does not allow for negative sources of evidence – evidence which disproves a hypotheses. The application domain of SIGMA and the very simple regions used as low-level evidence were the reasons why much of the conflict resolution could be performed in the high-level layer, but dealing with negative evidence becomes very important with complex scenes with many different object classes, as the space of potential interpretations becomes much larger. For the same reason, a more powerful bottom-up detection mechanism is needed in order to provide a good initialisation of the scene model. Because of this the middle layer should be able to reject hypotheses which conflict with the observations. The authors point out that a more general method is needed in order to deal with such cases.

There are many influential concepts used in SIGMA. Image interpretation is achieved via an interpretation cycle, which combines low-level detections and top-down hypotheses. Objects are described using appearance models, and the high-level context selects suitable low-level detector algorithms and focuses the processing on a specific region of the image. Also important is the fact that the partonomy and taxonomy is used to resolve conflicts, and to reason about the appearance of structured objects. Still, the part-whole reasoning step is demonstrated on simple structured models based on simple low-level primitives. This is completely understandable given the time when the work was performed, but more recent work on object detection has shown that more robust structured appearance models can improve object and classification performance, as will be shown in Section 2.3.

### 2.2.3 DyKnow

The problem of separating symbolic object representation from sensory inputs was tackled by a middleware component Dyknow [Heintz and Doherty, 2004], and used for guiding autonomous unmanned vehicles. The main task of this component is interfacing between different parts of the interpretation system, by providing a standard representation about the world, at different levels of abstraction. Such representation can help independent autonomous agents communicate and exchange knowledge [Heintz and Doherty, 2008].

The output of different parts of the system is represented as a set of *streams* at different abstraction levels. These streams can be low-level blobs with time-varying positions and features, or sequences of high-level queries about objects and events. DyKnow consists of *knowledge processes*, which process and combine these streams and produce new ones, at a higher-level of abstraction. The example application using car tracking [Heintz et al., 2007], uses a stream of car observations, which is processed by knowledge processes into a stream of currently active cars, a stream of car pairs, which is then used by a car relation computational unit which uses qualitative spatial relations between the cars to detect overtaking events. Each knowledge process accesses the data at a suitable level of abstraction.

Much of DyKnow's infrastructure deals with anchoring and processing of dynamic scenes, which is not applicable to interpreting static images, but it also performs classification of low-level evidence. The low-level input comes from an image processing module which delivers potential objects and their positions, based on tracked colour and thermal image sequences. These blobs are represented as "VisualObjects", which are manifestations of "WorldObjects", mirroring the distinction between "Views" and "Objects" which will be introduced in the next chapter. This separation makes it possible in principle to ignore faulty detections. The matching of a VisualObject to a WorldObject implies an interpretation of an observed phenomenon within the context of the scene, and assigning a class to a WorldObject is a classification step.

The low-level image processing layer delivers blobs representing visual phenomena after sensor fusion of several inputs, assuming that they will correspond to objects of interest (such as vehicles), and the system uses external knowledge sources such as GIS, for localisation and domain knowledge about roads. DyKnow has been used for automated vehicle navigation and tracking, where these assumptions hold. Feedback to the low-level vision processes is possible in principle, by formulating the low-level algorithms in the DyKnow formalism, but this has not been pursued[1]. The interpretation of complex scenes requires some sort of low-level error modelling and should address the more complex general classification and object detection problem.

While the system is generic and not specific to a domain, the issue of automatic learning of middle-level processes which generate knowledge (or classify evidence) has not been pursued by the authors, so modelling by hand is needed to adapt to new domains. While learning object models is not the central topic of this thesis, the challenges outlined in Section 1.1 suggest that learning is an important part of building a complete interpretation system, so a generic middle-layer should not be designed in a way which makes it difficult to use learning for adapting to new object classes and application domains.

### 2.2.4 Intermediate Level Ontologies

Creating knowledge bases for scene interpretation is difficult, since the design of the interpretation system involves modelling the input from the low-level algorithms, as well as modelling the high-level ontology which often involves specialised domain-specific knowledge. Specialised image retrieval schemes, like the one used for retrieving specific Qin dynasty statues [Soo et al., 2002] annotate images using a domain-specific ontology which makes it easy for an expert to formulate queries. But such systems depend on hand-annotation of images, which is a time-consuming process. An image interpretation system must be able to provide such descriptions automatically, and this requires good knowledge of both the specialised domain and the underlying image processing algorithms, something few experts possess.

[1]Personal communication, 2007

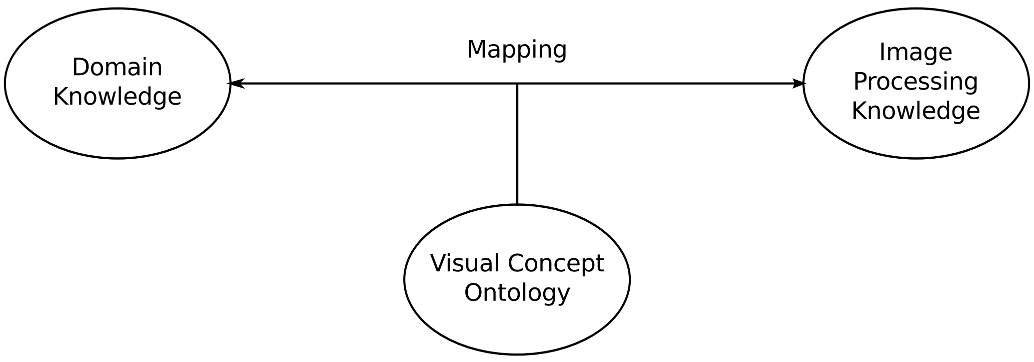


FIGURE 2.9: Schematic representation of a Visual Object Ontology, sitting between the high level (domain knowledge) and low level (image processing knowledge). Reproduced from [Maillot et al., 2004].

In order to decouple the construction of domain-specific knowledge bases from the implementation detail of low-level algorithms, Maillot and Thonnat [2008] proposed using an intermediate-level ontology of visual concepts. The proposed ontology is domain-independent, but expressed in terms of concepts which are easily understood, like elongation and colour. The motivation was to make it easy for domain experts to create domain-specific ontologies without using any image processing knowledge. The approach was successfully used to classify pollen grains [Maillot et al., 2003] and galaxies [Maillot et al., 2004], which usually require specialist knowledge.

The ontology consists of three parts: spatio-temporal concepts, texture concepts and colour concepts, which are all represented using Description Logics and applied to segmented objects and object parts. The spatio-temporal concepts include the shape of the object (e.g. “elongation”) and the spatial relations between parts (using RCC-8 [Randell et al., 1992]). The texture concepts define texture in terms of strength and regularity, and colour concepts use concepts such as hue and transparency. Each visual concept is associated with a specific classifier which detects whether the concept applies to the object or not. The final classification of the object is done by reasoning on the intermediate visual concepts. The advantage of this approach is that it can explain the classification decisions, unlike most machine learning approaches. This also makes the system more intuitive to an expert user.

The Visual Concept Ontology bridges the gap between low-level image and high-level concepts by describing the image in terms of intuitive concepts, which can be used to construct high-level knowledge bases more easily. But the calculation of the features which are used to define the visual concepts depends on correct segmentation of the object. The automatic segmentation of the input image remains a problem, and was either performed manually, or in domains where segmentation is easy. An interpretation system which aims at understanding complex scenes must also address the problem of object detection in crowded scenarios.

It should be pointed out that describing objects in terms of intuitive concepts is common in vision, but generally using a domain-specific vocabulary hand-tuned for the specific application. This work approached the problem from a generic point of view, using high-level knowledge representation and inference mechanisms.

### 2.2.5 Stochastic Grammars

The use of grammars for interpretation goes back to the early work in computer vision [Fu, 1982, Leyton, 1988]. This early work was hindered by the difficulty of modelling the complex visual world, and the lack of reliable symbols at the object level due to the difficulty of obtaining them using low-level image processing algorithms.

Recently, there has been a resurgence of interest in applying grammar-based approaches to vision, especially stochastic methods which make it possible to model noisy structures. At the same time, there has been a significant increase in computational power compared to the days of early vision work. Due to this, grammatical approaches have been extended down to the level of primitive visual tokens, modelling the appearance of complex objects using a grammatical formalism. This brings reasoning to the visual level, and has been shown to perform well for detecting certain classes of objects [Jin and Geman, 2006]. Essentially, the high-level knowledge was extended down towards the image level and object appearance is modelled in terms of stochastic grammars, effectively bridging the sense-reasoning gap.

In the context of scene understanding, the work of Zhu and Mumford [2006] is particularly interesting. They propose a comprehensive scene interpretation framework capable of segmentation and interpretation of scenes from different domains. The compositional structure of the scene is represented as an And-Or tree, which is extended with horizontal relations between the nodes. The relations are expressed as either hard (compatibility) or soft (statistical) constraints. At the root of the compositional tree is the scene, which decomposes into objects, object parts, and finally the visual primitives which make up the object parts in the image.

In order to construct object models, the architecture uses visual vocabularies at different levels of abstraction. Each element of the vocabulary is associated with a set of other elements through a set of possible "bonds". This makes it possible to connect visual primitives together to form more complex elements at a higher level of hierarchy. A typical example is joining several line segments at the ends in order to obtain a long, salient line.

At the lowest level, the primitives are taken from the dictionary of image primitives suggested by Guo et al. [2007], which roughly correspond to line segments, corners, T-junctions etc. They are likened to letters of an English word. At an intermediate level, these primitives are grouped into "graphlets", which are intermediate-level groupings, such as parallel lines, rectangles, extended curves, etc. These are learned statistically during training. Such low and middle-level primitives can be combined into salient object parts, such as a hand or a shoulder of a person. These form a domain-specific dictionary of object parts, which can also be connected to other parts or primitives.

The interesting aspect of this architecture from the middle layer point of view is not the grammatical formalism, but the middle-level modelling of object appearance. It enables a natural flow of top-down and bottom-up information. Since there is no strict separation between low-level image processing and high-level reasoning, standard stochastic grammar methods are used throughout.

The approach was successfully applied to several domains, with the focus on relatively well-structured objects such as bicycles, man-made buildings and shirts. Still, interpreting complex scenes remains a problem, as it requires an And-Or graph which accurately models the domain. At this time, the And-Or graph is modelled, not learned. The relations and parameters can be learned automatically from annotated images, but the annotations needed are detailed and time-consuming. There exists a database of 500,000 annotated objects, with the And-Or graphs available for 200 categories, which can be plugged into a high-level knowledge base, but scaling the process to thousands of categories needed for comprehensive image understanding remains a problem.

### 2.2.6 Blackboard Architectures

An early application of a blackboard architecture for modelling object appearance in interpretation was a part of the VISIONS system [Hanson and Riseman, 1978]. The knowledge of the scene is divided into long-term memory, modelling general knowledge about the shape and appearance of objects in the real world, and short-term memory, which is image-specific and describes the regions, surfaces and volumes detected in the image. Both representations are hierarchical. On the long-term side, the scene is decomposed into objects, which have certain 3D shapes (e.g. cylindrical or cubic), and 3D shapes consist of 2D surfaces which can be rectangular, circular, etc. On the short-term side, the objects decompose into volumes, surfaces, regions, edges, etc. Scene interpretation is performed as independent knowledge sources (KS) find correspondences between the observed evidence in the image (such as edges, line segments and regions) and the high-level models in the long-term memory. A search tree is traversed in order to navigate through the space of possible models. While the work of Chella et al. [1997] also used an intermediate description of the scene in terms of 3D primitives , their description was obtained in a purely bottom-up manner. The VISIONS system allows (at least in theory) for a more flexible mixture of bottom-up and top-down processing even below the level of 3D and 2D primitives.

The development of the VISIONS system brought about the need to apply goal-driven top-down control of low-level processes, leading to a middle layer component GOLDIE [Kohl et al., 1987]. GOLDIE consists of a set of independent agents (called *schemas*) which can perform image processing tasks such as region segmentation, line extraction and colinear line grouping on demand. The schemas produce low-level and mid-level tokens which can be grouped together by other schemas, always with a certain goal in mind (such as looking for a particular class of objects). The intermediate layers of the long-term memory contain knowledge about which features (such as colour or texture) are good for detecting certain classes of objects, so an attempt to locate a tree would be performed by the schema most likely to detect such features. A typical example is a top-down request to disambiguate between two possible labels for an image region – tree or sky. The system activates an instance of the region-segmentation schema and uses the knowledge from the long-term memory to find out which features best discriminate between the two classes. It then computes them on demand for the region in question.

The schemas communicate via a blackboard, where both goals and tokens are stored. The mid-level and low-level schemas are an extension of the high-level model that was successfully applied to the high-level reasoning layer of the VISIONS system, and as such represent an extension of high-level principles down towards the image level.

A more recent application of the blackboard system to image understanding was given by Guhl [2007]. The application domain was navigation for autonomous robots, but the principles can be applied to scene interpretation in general.

The architecture is based on a central blackboard system where the current understanding of the visual detections is represented. The knowledge needed for object detection is modelled as a set of independent software agents which take data stored on the blackboard, process it, and create new data. There are three types of software agents:

- **Low Level Processes**, which detect low-level phenomena, such as edges, corners and regions,
- **Knowledge Sources**, which know how to combine the low-level and mid-level detections into more complex shapes using Gestalt principles, and
- **Specialised Processes**, which detect complete objects, based on strong appearance models.

This flexible modelling allows the system to both use specialised low-level knowledge for detecting special classes of objects, when it is present, and to make a best interpretation of lower-level evidence to complete the understanding of the scene. Here, the Specialised Processes detect known objects (object recognition), and a combination of Low Level Processes and Knowledge Sources tries to discover and segment unknown objects (object detection), based on general, not class-specific knowledge about objects.

The visual detections are represented at three levels: raw sensor data level, feature level, and object level. The agents combine raw sensor data into features, combine features into more complex forms, and finally into whole object hypotheses in a distributed fashion, with different agents potentially contributing to the same hypothesis and improving the detection confidence. Such a system results in many hypotheses, including duplicated object detections and false positive detections in irrelevant areas of the image. A complete interpretation of the scene must reject some of them as false detections and choose the best of several possible detections in other cases.

The selection and combination of object hypotheses is done by collecting confidence values of all parts which contribute to a complete hypothesis and combining them into a set of hypothesis scores using a set of improper linear models. The exact equations for calculating scores were determined experimentally. While each of the individual scores correlates to the quality of a hypothesis on average, they all suffer from too many outliers, so similar hypotheses (selected as having similar outlines) at the same position are clustered together and their scores combined using another linear model to improve the robustness of object detection.

While the system allows for modelling disjunctions on the level of features (for a certain hypothesis to exist, a certain low-level feature must be present, and another one must not be present), it cannot express disjunctions at the level of hypotheses. This leads to problems in the cases where there are competing hypotheses at the same location, and only one of them is correct.

From the middle-layer point of view, the approach of combining smaller primitives into more complex visual tokens is similar to the approach of Zhu and Mumford, but whereas they use top-down modelling using stochastic grammars which model the structure hierarchically, the blackboard architecture facilitates a distributed process, combining different sources of knowledge. Like Zhu's approach, the best results are achieved for objects which can be detected by tracing and connecting edges and regions, and tested almost exclusively on man-made objects. This is understandable given the context of autonomous robot navigation in indoor environments, but can become a problem when trying to interpret complex natural scenes.

The parallel and distributed aggregation of low-level primitives into objects is evocative of the human vision process, which processes input at many different levels of abstraction simultaneously [Ullman, 1984], but it complicates the design. While the use of Specialised Processes makes it easy to plug in off-the-shelf algorithms for new domains, the general image processing knowledge needs to be expressed as Knowledge Sources, and the factors for the individual equations for combining confidences might also need to be re-adjusted. Since best equations and factors were determined experimentally, this process can be cumbersome and based on trial and error.

### 2.2.7 Related Architectures in Robotics

The problem of matching sensory input to high-level concepts also exists in robotics. Much of the work on the topic deals with planning, multi-sensor fusion and robot control, but some of the research focuses on interpreting the input from the visual modality and specifically with the issue of obtaining reliable object detection and classification from visual data. Typical examples are identifying objects which can be handled, and building a map of the surrounding environment for high-level planning.

While some approaches in robotics eschew symbolic representation and model behaviour using connectionist models, many robotic architectures are built around high-level planners and reasoning. An introduction to the signal-symbol matching problem in robotics is given by [Bajcsy and Kosecka, 1995], with an argument why symbolic reasoning is important.

The symbolic planning architectures in robotics owe much to the early work of Nilsson at al. dating back to the mid 1960s [Rosen and Nilsson, 1966, Fikes et al., 1972]. Their robot "Shakey" was the first robot capable of reasoning about its environment, and many algorithms spawned by the project are still influential to this day. A detailed overview of the architecture, including the vision routines, is given in [Nilsson, 1984]. The architecture provides a set of specialised vision routines which answer specific questions relevant for planning robot actions, such as locating the floor region, and locating potential obstacles by analysing the edges which separate regions from the ground region. The vision routines are understandably simple given the available processing power at the time, but it can be seen that the problem of mapping high-level requests to low-level vision algorithms was already seen as one of the central issues to be tackled.

Hexmoor et al. [1993] introduced an architecture which enables symbolic instructions to be mapped to robot sensors in order to use natural language commands instead of programming at a low level. Low-level vision is limited to describing objects through simple properties such as colour, and much of the architecture deals with controlling the robot movement. A more detailed discussion of the anchoring problem using the GLAIR architecture was given in [Shapiro and Ismail, 2003].

Hutber et al. [1994] introduced a middle layer between a perceptual multi-sensor system and an interpretation system in the context of the intelligent driver assistance system *PROLAB2*. In the bottom-up direction, the sensory input from a number of independent sensors is merged in a data fusion module, which produces a map of the surrounding, with all the detected obstacles. These are taken as input for the interpretation module, which determines the important obstacles and can use feedback to the sensory level in order to receive a refined map. The middle layer passes such requests to low-level in terms of Region of Interest (ROI) constraints, turning particular sensors on and off, and adjusting parameters.

Since the middle-layer interface holds the knowledge about handling the low-level sensors, the interpretation layer only needs to understand the relevant concepts (obstacles and zones) and need not worry about the raw sensory input. High-level requests are mapped to the sensors by a middle layer which encapsulates all the needed knowledge. The interpretation layer is only presented with information which is considered certain enough to be acted upon. However, the architecture does not deal with the classification of obstacles, or their role in the scene. Furthermore, the middle layer is hand-tuned to the specific sensor combination, which is advantageous for the specific application, but makes switching to new domains and sensor combinations more difficult.

Konolige et al. [1997] also match the low-level sensory information to a high-level representation, called "artifacts" in order to facilitate autonomous robot navigation. The sensory input is first grouped into "object hypotheses", which are then classified into artifacts. Route planning is calculated in this high-level space, without dealing directly with the sensory input. This enables easy formulation of tasks and thus the interaction with humans.

In order to detect artifacts, the low-level sonar readings are grouped into line segments, which are grouped into object hypotheses. These object hypotheses are then interpreted as artifacts, such as walls and doorways and tracked over time by maintaining the correspondences between artifacts and sensory input using the process called "anchoring", described later in this section. In addition to the sonar input, a specialised human detector module is also present, which delivers human hypotheses, which are also classified into artifacts and added to the current scene model. This means that there are several different sources of information, specialising in detecting different scene objects (or *artifacts*).

Schlegel et al. [1999] presented an architecture for autonomous robots which divides processing into three layers: the subsymbolic skill layer, sequencing/execution layer and the high-level deliberation layer. The execution layer contains an interface between the symbolic knowledge and vision algorithms. While the interface governing robot actions is sophisticated, the vision algorithms are simple from today's perspective. Objects are detected by applying colour segmentation and matching the blobs to the models based on simple predicates describing the known object classes. The predicates defining interesting colours can be passed to the segmentation algorithms to find blobs of specific colours, e.g. to localise a yellow trash can or a person in a blue shirt. In the case of following a particular person, an edge model of the person is constructed from the region of interest and matched to later detections using the Hausdorff distance, in order to avoid confusions with people wearing similar clothes.

A more advanced vision architecture was given by Bredeche et al. [2003], where pixels are grouped to superpixels at different granularity levels, and a *bag of features* representation learned on labelled images. Their algorithm learns distinctive pixel formations for different classes and uses them to track the object of interest. The effectiveness was demonstrated on three classes.

The matching problem in robotics has been called “perceptual anchoring” by researchers, and it deals not only with classifying low-level evidence and giving it a role within a scene, but also with maintaining correspondences. In this sense, it has as much to do with tracking as with object recognition.

### 2.2.8 Perceptual Anchoring

In [Coradeschi and Saffiotti, 2000], the authors propose a formal definition of the anchoring problem. They define anchors in terms of maintained correspondences between a symbol system $\Sigma$ and a perceptual system $\Xi$ through a set of perceptual signatures $\Gamma$.

The perceptual system $\Xi$ is represented by a set of percepts $\Pi = \{\pi_1, \pi_2, \ldots\}$ and a set of attributes $\Phi = \{\phi_1, \phi_2, \ldots\}$. The percepts from $\Pi$ correspond to evidence which is assumed to originate from one object in the scene. This means that the presence of a lower-level processing layer is assumed, which performs perceptual grouping of evidence. In robotics, this is typically facilitated by the dynamic nature of the scenes, allowing for motion-based grouping. The attributes from $\Phi$ add together to form a feature vector of measured low-level properties describing the evidence in terms of colour, shape, texture, etc.

The symbol system $\Sigma$ is represented by a set of individual symbols $\mathcal{X} = \{x_1, x_2, \ldots\}$ and a set of predicates $\mathcal{P} = \{p_1, p_2, \ldots\}$. The predicates from $\mathcal{P}$ are logical symbols representing the low-level appearance of an object in the scene, and are similar to the visual concept ontologies from [Maillot et al., 2003].

Finally, the set of perceptual signatures $\Gamma = \{\gamma_1, \gamma_2, \ldots\}$ maps a combination of unary predicates from $\sigma \in 2^{\mathcal{P}}$ to an expected set of low-level values from $\Phi$, and is used to pass appearance expectations to the low level. For example, a robot might try to distinguish between a small and a large object, or between a red and a blue car.

These terms are used to define a perceptual anchor. An anchor is the internal representation of the object in the scene at any given time. An object $\mathbf{o}$ is represented by a high-level symbol denoting $\mathbf{o}$, a percept generated by observing $\mathbf{o}$, and a perceptual signature representing the appearance of $\mathbf{o}$. An example of matching a detection $\pi_1$ with a perceptual signature $\gamma_1$ to the high-level symbol “car-1” is written as:

$$\alpha : t_1 \mapsto \langle \text{car-1}, \pi_1, \gamma_1 \rangle \tag{2.1}$$

As was pointed by the authors themselves, this architecture is more concerned with maintaining correspondences between percepts and symbols throughout the evolving scene than with the object detection and classification. The crisp classification based on attribute value ranges is suitable for distinguishing between two classified objects (e.g. between a small red car and a large blue car), but not for object detection and classification in complex realistic scenes, where state of the art approaches build complex probabilistic models which can reason about object parts under the presence of occlusion and clutter.

While this architecture might be too simple for the generic interpretation of complex scenes, it offers a clean formalism for expressing correspondences between sensor signals and a symbol system. It also offers a compressed description of an object in terms of attribute value ranges. Although this description is not discriminative enough for bottom-up classification in the general case, it can be seen as a useful way to pass top-down feedback to the lower layers of a scene interpretation system.

## 2.3 Categorisation and Object Detection

Much of recent work in Computer Vision has concentrated on object detection in complex scenes, under a wide variation of illumination and pose. A number of different approaches have been applied to standard benchmarks, such as the TU Darmstadt Database [Leibe et al., 2004], UIUC Car Database [Agarwal et al., 2002], the VOC Challenge datasets [Everingham, 2006], and the Caltech [Fergus and Perona, 2003], MIT-CSAIL [Torralba et al., 2003] and TU Graz [Opelt et al., 2006] datasets. The approaches range from boosting with a small set of object-independent features [Viola and Jones, 2004] to full probabilistic models of object shape [Fergus et al., 2007]. What is common is the modelling of objects based on smaller parts, which are matched to a small codebook of representative parts, and which can be reliably detected. Parts are typically represented by image patches centered around interest points or regions detected by a segmentation algorithm. The model captures typical spatial arrangements of these parts, resulting in increased robustness with respect to intra-category variation, occlusion and illumination. The modelling of objects as compositions of salient parts is biologically motivated, and there is evidence that human vision system derives some of its robustness from such representations [Palmer, 1977, Biederman, 1987, Wachsmuth et al., 1994, Logothetis and Sheinberg, 1996].

Approaches to model-based object recognition generally classify a whole image (or a sub-window), and an object detection strategy is needed in order to find all objects in a scene. Detection strategies depend on the algorithm used and include sliding windows, Hough voting or iterative methods. In the end, the scene is given a category label, and the objects in the image are located and classified.

The categorisation approaches have many things in common with scene interpretation. They group evidence into semantically meaningful groups representing objects and assign labels to them from a high-level ontology, thus giving a meaning to parts of the image. They can recognise objects from dozens of visually complex classes in complex, crowded images. They often use object hypotheses to refine the search for fitting evidence based on an object appearance model.

Categorisation approaches generally give a label to the entire image (which is a basic form of image interpretation), but they rarely model relationships between individual objects. Even the challenging scenarios from the PASCAL dataset [Everingham, 2006] typically feature a small number of objects. The challenge is in finding occluded and poorly visible objects in complex scenes, rather than understanding the complex scene in terms of relations between the individual objects. Because of this, object detection and categorisation can not serve as a replacement for high-level scene interpretation, but it would be useful to use a state of the art algorithm for detecting objects in a complete scene interpretation system.

The middle-layer approaches described in the previous section have typically modelled the object appearance hierarchically, starting from edges and corners, or relied on a low-level process to deliver potential objects which were then classified based on simple features, or the objects' role in the scene. At the same time, patch-based object detection and categorisation techniques have become prevalent in recent years due to their performance on a wide variety of tasks and ability to learn hundreds of classes from limited annotation. In addition to detecting objects, many algorithms deliver a quality measure, which can be used to reject poor detections.

Of course, one can always use an object detector which provides a quality measure as a module, and pass the detections to a high level module, as used by many systems described in Section 2.2.1. It is often even desirable to do so, due to the effectiveness of certain algorithms for detecting certain classes of objects. However, experience from past systems suggests that it is beneficial to allow the scene context to flow into detection and classification decisions, as done by Guhl [2007] and Zhu and Mumford [2006], rather than to rely on specialised detectors to detect all important objects in a scene in a purely bottom-up fashion.

This section will present an overview of important recent work on object detection which is interesting for designing a middle-level interface in a complete interpretation system.

### 2.3.1 Wholistic Approaches

Although flexible and hierarchical arrangements of image patches are particularly interesting for object detection and classification in a complete interpretation loop, many successful approaches to object detection compute a global descriptor of the image (or a window) in order to classify the object. Wholistic approaches typically slide a window of fixed size across the image and use the contents of the window (or a numerical descriptor derived from it) to classify the contents of the window. Binary classifiers are generally used, so they have to be learned separately for each class.

Early approaches include the face detector of Sung and Poggio [1995], which learns a set of prototype clusters for the face and non-face classes and uses a novel distance measure for comparing the contents of the window with the model, and Rowley et al. [1996], who trained a neural network to classify an image window. Recent approaches improve classification robustness using subspace methods such as [Fidler et al., 2006].

Another way to classify objects is to extract a set of interest points, match them to a codebook, and classify using the set of features, discarding the spatial relation between them. Such "bag of words" approaches are used in language processing [Joachims, 1998], and were adapted to the image domain by some researchers [Zhu et al., 2002, Csurka et al., 2004, Nowak et al., 2006].

A crude way to model the shape of the object is to divide the image into a rectangular grid and use histograms to capture image variation within the blocks. Such approaches include the Histogram of Oriented Gradients [Dalal and Triggs, 2005] and the pyramid scheme of [Lazebnik et al., 2004].

Despite the improvements in part-based visual models in recent years, wholistic approaches are still remarkably effective in comparison. Although they operate in a purely bottom-up fashion, they often outperform more complex flexible models. One possible reason for this is insufficient use of high-level information and global scene context during object detection which could improve detection accuracy.

### 2.3.2 Flexible Arrangements of Parts

Many state of the art object recognition algorithms model object appearance as a flexible arrangement of parts. They differ in how the object parts are represented and how the relations between the parts are modelled. One major approach to modelling parts is to manually select a visual vocabulary. This approach is biologically inspired [Julesz, 1981, Marr, 1983] and includes Haar-like features [Papageorgiou et al., 1998], textons [Shotton et al., 2009] and image primitives [Guo et al., 2007]. Another approach is to learn common image elements by clustering patches gathered around interest points, as done by [Leibe and Schiele, 2003, Ommer and Buhmann, 2010]. Most approaches use weighted sums or probabilistic distributions to determine whether the combination of the parts represents the object being sought.

Weighted sum methods are fast to evaluate, making them popular in real-time applications, such as face-detection in videos. Boosting is commonly used to learn weights for the individual parts. Viola and Jones [2004] presented an influential approach, which uses Haar-like features as parts (see Figure 2.10). The features are efficiently computed

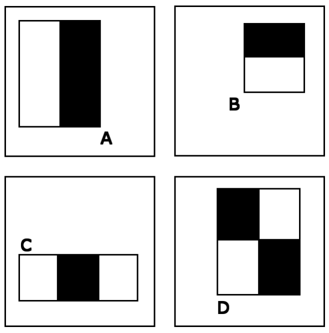


FIGURE 2.10: Rectangular features used for face detection by [Viola and Jones, 2004].

by using an integral image which speeds up the summing up of pixel values in rectangular regions. Recognition is performed by a cascade of increasingly complex classifiers (using 32 stages). The first few stages already discard most of the non-face detections, and each further stage discards further false-positives until only face detections are left. This design reduces the computational complexity as most false positives are discarded early in the pipeline, using simpler classifiers.

The approach is applicable to object classes other than faces, and was used by many researchers [Šochman and Matas, 2009]. However, for detecting a number of objects from many different categories, one needs to train a special detector for each class, run them in sequence, and finally resolve potential conflicting detections. Torralba et al. [2004] proposed an approach which uses multiclass boosting instead of many separate classifiers. In their framework, they allow weak classifiers (and thus features) to be shared between different category classifiers, and demonstrate the effectiveness on 21 object classes.

Agarwal and Roth [2002] presented an approach not based on boosting. They use a Förstner detector to find interest points and extract patches around them. The patches are clustered into a codebook, using normalized correlation as a distance measure. Then, a binary feature vector of the image is constructed, with each bit encoding a particular number of occurrences of a particular codebook entries, or the number of occurrences of a particular relation between two entries. The resulting feature vector is very long, but sparse, leading to efficient learning. The approach does not need hand segmentations, only positive and negative examples, but the results improve significantly if the training and testing images are cropped to the bounding box of the object.

Early approaches to modelling spatial arrangements of object parts focussed on deforming a rigid template. The early work of Yuille [1991] on face recognition used gradient descent on a deformation energy function. It was followed by the work of Lades et al. [1993] who linked landmark points on the surface of objects and Cootes and Taylor [1996], who model global deformations using Eigenspace methods. Finally, Amit and Geman [1998] scale images to a reference grid, in which the spatial relations between local image parts are defined.

In the meantime, the focus seems to have shifted towards modelling part configurations explicitly. Leibe and Schiele [2003] expand on Agarwal and Roth's patch-based approach by extracting 25x25 pixel patches around Harris corners and using agglomerative clustering to construct a codebook of patches. A distance measure is defined using Normalised Greyscale Correlation and each detected image patch is matched to several codebook entries, with the distance below a threshold. Each codebook entry remembers all positions it was activated in (relative to the object centre) and a segmentation mask for each of the positions. The position is used to locate the object using a Hough voting scheme, and the segmentation mask is used to derive a soft segmentation of the object once it has been recognised.

The model estimates the probability $p(o_n, x|e, l)$ of the object $o_n$ at a location $x$ given a patch $e$ at location $l$ as sums up over all patches and positions to obtain an object score. The approach was applied to a number of object classes, including pedestrians, cars and cows. The ability to provide segmentation comes at a price of needing segmented images for learning. Borenstein and Ullman [2008] present a combined bottom-up and top-down classification and segmentation scheme, which is also based on image fragments, but is capable of learning the fragment segmentation masks automatically from unsegmented images.

A similar approach was proposed by Gu et al. [2009], by using segmented image regions instead of patches. The image is first segmented using a hierarchical region segmenter, and each region is characterised by a feature vector representing shape, colour and texture. Each object is therefore represented by a set of regions described by feature vectors (the regions are treated as a bag, with hierarchical relations discarded). During a verification step, each region from an exemplar image $\mathcal{I}$ is compared with each region from the query image $\mathcal{J}$ and the smallest distance is recorded for each region from $\mathcal{I}$. The distance between $\mathcal{I}$ and $\mathcal{J}$ is defined as a weighted sum of these minimum distances, with weights being learned from random pairs of images from the training set. The effectiveness is demonstrated through state of the art results on several benchmarks.

As with Leibe et al., the individual regions vote for the object centre after being matched to the most similar region seen during the training stage. The power of the region approach derives from the fact that the exact boundaries of the object are known, and after image regions are marked as either belonging or not belonging to the object, a top-down segmentation step can extract the exact contours of the object.

Another approach [Felzenszwalb and Huttenlocher, 2003] expands on the very early work on pictorial structures of Fischler and Elschlager [1973] and represents objects as a combination of parts connected through springs. Matching is done through an efficient approximation of the energy minimisation problem. One advantage of their formulation is that it can model articulated objects such as human bodies.

An influential probabilistic model of deformable objects was presented by Burl et al. [1998]. The human face is represented by five manually selected parts (eyes, nose tip and mouth corners), and these parts are located in new images using a correlation-based method. The learning algorithm learns the full JPD of the object part locations $p(X)$. Any possible value of $X$ is a face hypothesis, which can be evaluated using a score function which involves both the sum of shape log-likelihood ratio and the response of the part detectors. This formulation allows for a tradeoff between the quality of the shape and the quality of individual parts. Finding the best hypothesis is difficult, as there are two factors one wishes to maximise. The authors solve the problem by initialising hypotheses using pairs of detected features (hypothesising the remaining object parts) and then using a gradient descent approach to find the local optimum match.

Unlike previous approaches, the model is deformable, i.e. it can accept poorly matched parts if the resulting shape fits better, and it looks for the best matching object hypothesis given these two conflicting goals. The performance approach was evaluated on frontal images of faces. While the learned shape model is scale-invariant, learning needs to be performed on similar images at the same scale.

The approach was expanded on by Weber et al. [2000], who use learning for both the typical object parts (thus forming a codebook of typical patches) and the important configurations of parts (the object shape), both of which were hand-selected by Burl et al. During the learning process, interest points are found using the Förstner operator and clustered into a codebook using k-means, and model parameters are learned using Expectation Maximisation. The approach was tested on images of cars and faces.

Fergus et al. [2007] expand on the work of Burl and Weber in a few important ways: they use an interest point detector of Kadir and Brady [2001], they learn appearance simultaneously with shape and model appearance variation within the class, and their models are scale-invariant. The algorithm is general, needs only weak supervision, and performs well in practice, but is restricted by the fact that learning time is exponentially related to the number of parts in a model, leading to models consisting of only 3 to 7 parts.

Finally, Fei-Fei et al. [2006] provide an approach for one-shot learning of categories based on the previous work by Fergus and Weber. After learning some categories the traditional way, new categories can leverage this knowledge to learn new categories faster. They demonstrated promising results using only 1-5 training images per category, as opposed to hundreds or thousands needed by other approaches.

### 2.3.3 Hierarchical Approaches

Categorisation (and object detection for scene interpretation) attempts to distinguish between objects belonging to many different categories. This requires both discriminative power, as well as a tolerance for large intra-class variation and the variation with respect to illumination, occlusion and pose. As pointed out by Thureson and Carlsson [2004], increasing the descriptiveness of image parts calls for larger patches, while increasing the tolerance to variation calls for smaller ones. Thureson's answer was to focus on capturing qualitative relations between triplets of pixels in a histogram instead of modelling the objects in terms of patches. At the same time, this realisation has led into research on hierarchical object appearance models.

Schmid [2001] proposed a more general approach consisting of two stages. During the first stage, a greyscale descriptor is calculated for each pixel in the image by convolving the image with Gabor-like filters. The descriptors are probabilistically matched to a codebook of generic descriptors (determined by k-means clustering) based on distance. Then the co-occurrence of generic descriptors in the image is captured in "spatial frequency clusters", which are represented by joint probabilities of their frequencies over neighbourhoods. The spatial arrangement of the parts is not taken into account, only the distance to the neighbourhood centre, making these compositions of patches rotationally symmetric. The two-stage modelling increases the distinctiveness of the detected object parts.

More recently, a graphical model of Bouchard and Triggs [2005] represents objects as collections of parts which, in turn, consist of one or more patches from a codebook. The assignments of patches to object parts are *soft*, and modelled by assignment probabilities learned during training. The training uses Expectation Maximisation to learn the position and scale variance (represented by Gaussians) and the vector of parent assignment probabilities. Hough voting is used to match the model to an image. Scalzo and Piater [2005] also learn a hierarchical graphical model representing the positional constraints between parts using Expectation Maximisation and use Belief Propagation to detect the presence of higher-level parts in the image.

Hierarchical features were also explored in the approach of Epshtein and Ullman [2005]. It starts by using normalised cross-correlation to find a set of useful patches at different scales. It then iteratively subdivides each patch into sub-patches, by comparing positive and negative examples of a class from different images where the patch was observed, and keeping sub-patches which discriminate well between classes until a hierarchy of patches is obtained. During recognition, the response of each sub-patch is weighted and summed up, and this is iteratively repeated with the parent patches until the top-most node is reached. The weights are learned using a network model.

Whereas Epshtein decomposes larger patches into smaller ones, Fidler and Leonardis [2007] start with small, universal features at the lowest level and learn flexible arrangements over several layers of hierarchy. At the lowest level, the features are found by extracting local maxima of the Gabor energy function. Object *parts* consist of features (or other parts) in a certain spatial arrangement, modelled as offsets from a central part using Gaussian distributions, after accounting for the central part's orientation. This provides a hierarchical representation which is orientation and scale-invariant. The lower layers of the hierarchy are category-independent, and the higher layers focus on category-specific features, allowing for incremental learning of multiple categories.

A model not based on patches or edge-like features was proposed by Todorovic and Ahuja [2008]. They segment an image hierarchically, giving a segmentation tree. Segmentation trees from images belonging to the same category are matched in order to find commonly ocurring subtrees, which are interpreted as instances of the object category, or their parts. These subtrees are fused into a canonical graph which models the intra-category variability and serves as the category model. Subtrees were also matched by Drauschke [2009].

Huu et al. [2005] present a biologically inspired approach based on Fukushima's Neocognitron [Fukushima, 2003]. Recognition is based on units called *maps*, which are sets of local classifiers (*units*) arranged in a way that neighbouring units process neighbouring visual stimuli. Maps are organised hierarchically, with each layer consisting of maps which take the output of the previous layers as input, essentially building a multi-layer neural network. The first layer of maps takes pixels as inputs, the next layer can compute edges, and each subsequent layer is trained to detect increasingly complex structures, such as eyes, mouth, and finally the whole face. The approach demonstrated the ability to detect faces of different people in different postures, and even interpret a simple scene.

#### 2.3.3.1 Ommer: Hieararchical Visual Categories

The approach of Ommer and Buhmann [2010] is explained in more detail, since it has been adapted to fit into the middle-layer framework presented in this thesis. The authors suggest a hierarchical scheme which starts at the level of universal image patches, which are aggregated into visually salient compositions and higher-order compositions, building a hierarchy of compositions which reaches up to the object level. This hierarchical approach is appealing since it mirrors the hierarchy above the object level, which is inherent in knowledge-based scene interpretation. In other words, it extends the compositional approach used by interpretation systems such as Bayesian Compositional Hierarchies and KONWERK down to the primitive patch level. This section briefly describes the recognition process, which is explained in more detail in Section 3.3.3.1.

The algorithm starts by detecting interest points at different scales using a Harris detector, and describing image patches (called *atomic parts* around interest points using 40-dimensional histograms of edge orientation, edge strength, and colour. The 40-dimensional histograms are clustered into a codebook of class-independent $k$ prototypes using k-means. Each detected interest point is then described by a k-dimensional descriptor, where each dimension is a Gibbs probability representing the similarity of the patch to one of the codebook prototypes.

From the set of all atomic parts, a set of 40 is selected at each scale, and grouped together with nearby parts using simple proximity grouping. Each one of these *compositions* is described as the average of the descriptors from all the grouped parts. The compositions are therefore also represented by k-dimensional vectors. Furthermore, random tuples of compositions are sampled, representing related object parts.

The recognition starts by using relevance functions which discard irrelevant compositions and higher-order compositions of compositions. The remaining compositions enter into a Bayes network which combines the individual class probabilities conditioned on the parts and their locations into a complete probabilistic model of object appearance. The relevance functions and the class probabilities are all learned automatically during a training stage, which is not described in detail here.

Despite being very generic and being able to learn from minimally annotated images, this approach showed state-of-the-art performance on the challenging Caltech-101 [Fergus and Perona, 2003] database, making it a very interesting approach for generic scene interpretation. An important aspect of this approach is that the patch codebook is class-independent, as in [Fidler and Leonardis, 2007]. This allows for robust statistics at the lowest level, but with a more descriptive repertoire than offered by Haar-like features. At the same time, the compositional modelling at higher levels ensures sufficient descriptiveness even for hundreds of classes. The basic elements used for recognition are compositions of patches, which are described as a set of Gibbsian probabilities, improving robustness. In this sense, the approach is both general and discriminative, which is important for generic image interpretation.

Unfortunately, the modelling of compositions as probabilities makes it difficult to localise objects using Hough voting, as used in many competing approaches. The compositions are expressed as vectors of values and are not matched to exemplars seen during learning. This means that the expected offset from the object centroid for a composition $g_j$ is modelled by the probability $p(s_j|g_j)$, which is difficult to estimate in practice. The approach used by the authors for estimating the object centre is a complex iterative process which assumes that there is a maximum of one instance of any known object class in the image. Since this assumption does not hold in a realistic image interpretation scenario, a different localisation method is needed, such as sliding windows or random sampling.

The grouping of small visual primitives into a hierarchy of compositions is similar in principle to the approach of Zhu and Mumford [2006]. However, whereas Zhu constructs the hierarchy in terms of a logical structure and learns the constraints, this approach learns the relevant compositions automatically. In contrast to Zhu's approach, which extends a high-level formalism down to the level of visual primitives, here the low-level probabilistic model is extended up to the level of objects. Unlike [Fidler and Leonardis, 2007], the model is probabilistic.

#### 2.3.3.2 Towards Scene Modelling

Sudderth et al. [2005] presented an approach that extends the concept of flexible configurations of parts beyond the object level, thus modelling the entire scene. Li et al. [2009] expand on this approach by adding regions, and by using tag annotations common in internet photo galleries. The result is an ambitious system capable of automatic scene segmentation, annotation and classification. The objects in the scene are represented by both regions and patches. The regions are characterised by shape, location, colour and texture, each feature type quantised into a set of codewords. The patches are clustered into a codebook by vector quantising SIFT features. In order to obtain sufficient training data, the algorithm can learn from tagged images downloaded from the internet, where tags serve as crude, noisy annotation.

The probabilities of configurations of objects, regions, pathches and tags are modelled by multinomial distributions The scene is represented by a large JPD:

$$p(C, O, R, X, S, T, Z|\eta, \alpha, \beta, \gamma, \theta, \phi), \tag{2.2}$$

which factors into simpler distribution under certain independence assumptions. In the equation above, $C$ is the classification of the entire scene, $O$ are the objects in the scene, $R$ are the regions, $X$ are the patches, $T$ are annotation tags, and $Z$ is a region index which relates image regions to tags. The remaining variables define the multinomial distributions which are used to sample the objects, patches, regions, and other variables of the model.

The variables of the model are automatically learned. First, a set of appropriately tagged images for each category is downloaded from flickr.com. From these, a set of representative regions is learned, which is used to locate seed regions belonging to the object in a new image, giving a set of seeds and their corresponding labels. Such partially annotated images are used to learn the full model using collapsed Gibbs sampling.

Of all the categorisation approaches presented in this section, this one comes closest to the goal of scene interpretation, as it models the complete scene using co-ocurrences of semantic object categories. Particularly interesting is the ability to learn from loosely annotated and even unlabelled data. However, the scene is modelled in terms of co-occurrence of objects, not in terms of more complex relations between them, which offers a crude interpretation of the scene: scene category and the detected objects. The earlier work by [Sudderth et al., 2005] uses a more complex model of the scene which includes the spatial relations between objects, but even this formulation is difficult to extend to complex hierarchical scenes, especially scenes involving time and more complex knowledge-based constraints.

Finally, Rabinovich et al. [2007] introduce a way to combine contextual information into a probabilistic object classification scheme. The image is segmented into large regions, and for each region $S_n$, $n \in \{1 \ldots k\}$, a bag-of-features representation is calculated, and a class probability $p(c_i|S_i)$ estimated as a bottom-up classification of the region. The segments themselves are fed into a Conditional Random Field similar to [Heesch and Petrou, 2009] and provide the *interaction potentials* $\phi(c_i, c_j)$ for every combination of $c_i$ and $c_j$. The complete classification of the segments is given by

$$p(c_1 \ldots c_k|S_1 \ldots S_k) = \frac{exp\left(\sum_{i,j=1}^{k} \phi(c_i, c_j)\right) \prod_{i=1}^{k} p(c_i|S_i)}{Z(\phi, S_1 \ldots S_k)}, \tag{2.3}$$

where $Z(\cdots)$ is the partition function. The authors demonstrate that the approach improves the classification on the PASCAL dataset, and it can be applied to any classification scheme which provides probabilities.

A number of other approaches use scene context to improve object detection and classification, such as [Marszalek et al., 2009]. They are not as interesting from a middle-layer perspective, but are noted here as a part of a growing trend towards scene modelling and context propagation in recent work on object recognition.

## 2.4 Summary

There is a definite trend in the field of object detection and categorisation towards more flexible and hierarchical models which can learn and detect hundreds of categories [Ommer and Buhmann, 2010]. Object detection and categorisation algorithms are getting more complex and are starting to model aspects of the scene which are usually associated with high-level vision, such as using co-occurrences of object categories in the image [Li et al., 2009], but at this level they begin to suffer from the lack of expressiveness of the probabilistic formulation used and the increasing need for learning data.

At the same time, recent middle-layer architectures borrow from state of the art in object recognition to create more advanced appearance models instead of relying on accurate object detections from black-box detectors. State-of-the-art architectures show that classical high-level models can not easily express the extremely noisy and varied detections at the image level, leading to stochastic constraints [Zhu and Mumford, 2006] or complex hand-tuned models [Guhl, 2007] at the level of image primitives. Such approaches facilitate the top-down and bottom-up flow of information, but do not match the state-of-the-art algorithms in terms of performance or learning efficiency.

While both the low-level and high-level camps have come a long way towards complete scene understanding by extending their respective frameworks, the two communities have mostly worked in parallel. As pointed out in [Maillot et al., 2004], few people possess the expertise in all relevant aspects of image understanding. This thesis addresses the gap between low-level image processing and high-level interpretation by proposing a separate middle-layer component, which defines a clear interface allowing for propagation of uncertainty and contextual information. Such a separate architecture is able to re-use much of the existing state of the art knowledge and implementations, potentially resulting in better interpretation systems.

The following chapter discusses the requirements for such a component and introduces a new architecture capable of easy integration with existing low-level and high-level algorithms, but still allows for error modelling and top-down and bottom-up information flow.

# Chapter 3

# Middle-Layer Architecture

The previous chapter has outlined the challenges relating to automatic understanding of static images. This chapter presents a generic image interpretation architecture, in which these challenges are tackled by a separate middle-layer component. As shown in Figure 3.1, a complete interpretation system is divided into a low-level part (the image processing modules), a high-level part (the high-level system) and a middle layer connecting the two.

Based on this, we define the tasks that a middle layer should perform in a generic scene interpretation system:

- Low-level integration: selecting and controlling low-level algorithms
- High-level integration: providing a standardised interface to a high-level interpretation system

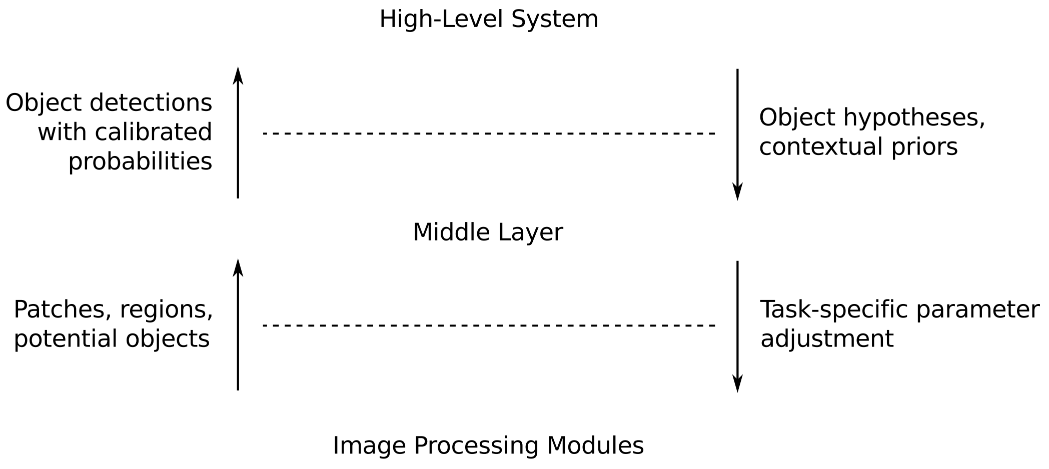

High-Level System
Object detections with calibrated probabilities
Object hypotheses, contextual priors
Middle Layer
Patches, regions, potential objects
Task-specific parameter adjustment
Image Processing Modules

Figure 3.1: Information flow in the proposed middle-layer architecture.

- Classification: integrating evidence into an evolving interpretation. Typically, this involves both determining the class of a detected object (e.g. “window”) and the role this object plays in the high-level model of the scene (“the middle window in the top row”).
- Top-down feedback: using the scene context to focus and parametrise image processing, and to disambiguate between similar classes
- Conflict management: resolving conflicts between low-level observations and high-level expectations
- Bookkeeping: keeping track of previous assignments in order to facilitate the step-wise process

The middle layer has a mediating role, and steers the course of interpretation by assigning evidence to high-level models. As additional evidence gets assigned, some potential scene explanations have to be discarded, and others refined, leading to an improved understanding of the scene.

In this chapter, a middle-layer architecture capable of performing these tasks is presented. The proposed architecture focuses on the integration and classification aspects, as well as providing top-down feedback for improved classification. Basic infrastructure for resolving conflicts and bookkeeping is introduced, but these topics are not explored in detail. A generic approach to the role assignment problem is presented for probabilistic systems.

## 3.1 Definitions and Key Design Decisions

As shown in Figure 3.1, the middle layer is seen as a separate part of a complete interpretation system, facilitating the information exchange between two separate layers.

- **Image processing modules (IPM)** take raw image as input and detect salient regions and interest points.
- **High-level system (HLS)** reasons about the objects in the scene and provides context for improving the detection and classification of low-level evidence.

The architecture allows for different image processing modules to be easily integrated and used together, and allows for grouping smaller regions and interesting patches into potential object detections. The high-level system can be probabilistic or crisp in nature. The integration of IPMs and HLS is described in more detail in Sections 3.3 and 3.4, respectively. Most of the interpretation experiments in Chapter 5 used low-level evidence roughly corresponding to individual objects in the image, but the architecture also allows for grouping object parts using structured appearance models, as shown in Section 3.3.3.

At the moment, no single algorithm is able to accurately detect all types of objects one might wish to interpret, or even all types of objects in a given scene. Although there are interpretation systems capable of extending the high-level hierarchy all the way to the level of individual edgels and texture elements [Zhu and Mumford, 2006], the dependence on the assumptions about the low-level processes make such systems difficult to adapt to radically different domains. The design based on a separation between image processing modules and the high-level system makes it possible to combine off-the-shelf algorithms with advanced reasoning systems, making it possible to adapt to new domains more easily.

Figure 3.2 shows the middle-layer architecture and introduces the main concepts:

- **Evidence** is the output of image processing modules. A piece of evidence consists of a bounding polygon and a feature vector describing the covered region. Typical evidence includes image regions, interest points, or even results from a specialised object detector. The evidence can be grouped together to form a hierarchy, resulting in **Structured Evidence**. Evidence which corresponds to a complete object, such as a group of interest points, is also referred to as a **Detection**.
- **View** represents an observation of an object in the image. Its class and role in the scene are defined by the scene object it represents, and its appearance is represented by the evidence it is matched to.
- **Scene Object** is a high-level concept used by the high-level system to reason about the scene. The information describing a Scene Object is dependent on the HLS used and the domain, and typically includes 3D position, common-sense knowledge, and other information about the object in the given domain, such as probabilistic constraints.

The middle layer interacts with the image processing modules by receiving *evidence* and information about how to group and classify them. It interacts with the high-level system by exchanging *views*.

The central task of the proposed middle layer is **matching** low-level evidence to high-level scene objects by grouping evidence and assigning it to views. This occurs either bottom-up, by looking for evidence which is indicative of an object and creating a view, or top-down, by accepting a **hypothesis** from the high-level system, which indicates where an object is expected in the image. A scene object is said to be **confirmed** if it is matched to evidence through a view. A view which is not matched to evidence is called a **hypothesis**. Therefore, the process of matching scene objects to evidence corresponds to the task of grounding high-level symbols.

Figure 3.1 shows the information flow in the proposed architecture. The middle layer receives low-level data, which are either object detections from a specialised detector or lower-level primitives such as patches and regions, and provides classified object views to the high level together with calibrated probabilities which describe how reliable the detections and classifications are. The HLS provides predictions and updated priors which can improve the classification and detection tasks in the middle layer. The IPMs can be controlled via generic parameters such as sensitivity, ROI and descriptions of the objects the system is expecting to find. This three-layered approach allows for expressing uncertainty and using high-level context to improve detection and classification, without introducing dependencies between a particular high-level system and any particular image processing module.

### 3.1.1 Rationale for Separating Appearance and Structure

In a natural scene, properties of objects are related to properties of other objects, but typically many properties are independent of each other. A major feature of the proposed architecture is that it separates the context-dependent properties of objects (such as size and position) from properties taken to be context-independent (such as colour or shape). The local appearance is represented by views, and the structural information by scene objects and relations between them. Since this decision is at the heart of the proposed architecture and introduces a number of consequences, a short rationale for it is presented. Section 3.2 will introduce the probabilistic foundation and show how probabilistic context can be used to improve classification, and introduce some of the consequences of this simplification.

There are three major reasons why the object description (in terms of evidence) is split in this way:

- Not all the correlations between all object features have to be modelled, reducing the complexity of the high-level model.
- Since feature vectors describing evidence are closely related to the low-level image processing modules which produced them, separating the knowledge about evidence produced by IPM makes the high-level knowledge base more general.
- Many state-of-the-art object recognition algorithms are based on local appearance of objects, and not their absolute position in the image or size, so they can be easily integrated into an interpretation system.

Theoretically, any split of the evidence vector is possible, as long as the two parts are statistically independent. In this thesis and all the presented experiments, the context-dependent part consists of position, class and size. Everything else is taken to describe context-independent local appearance of objects.

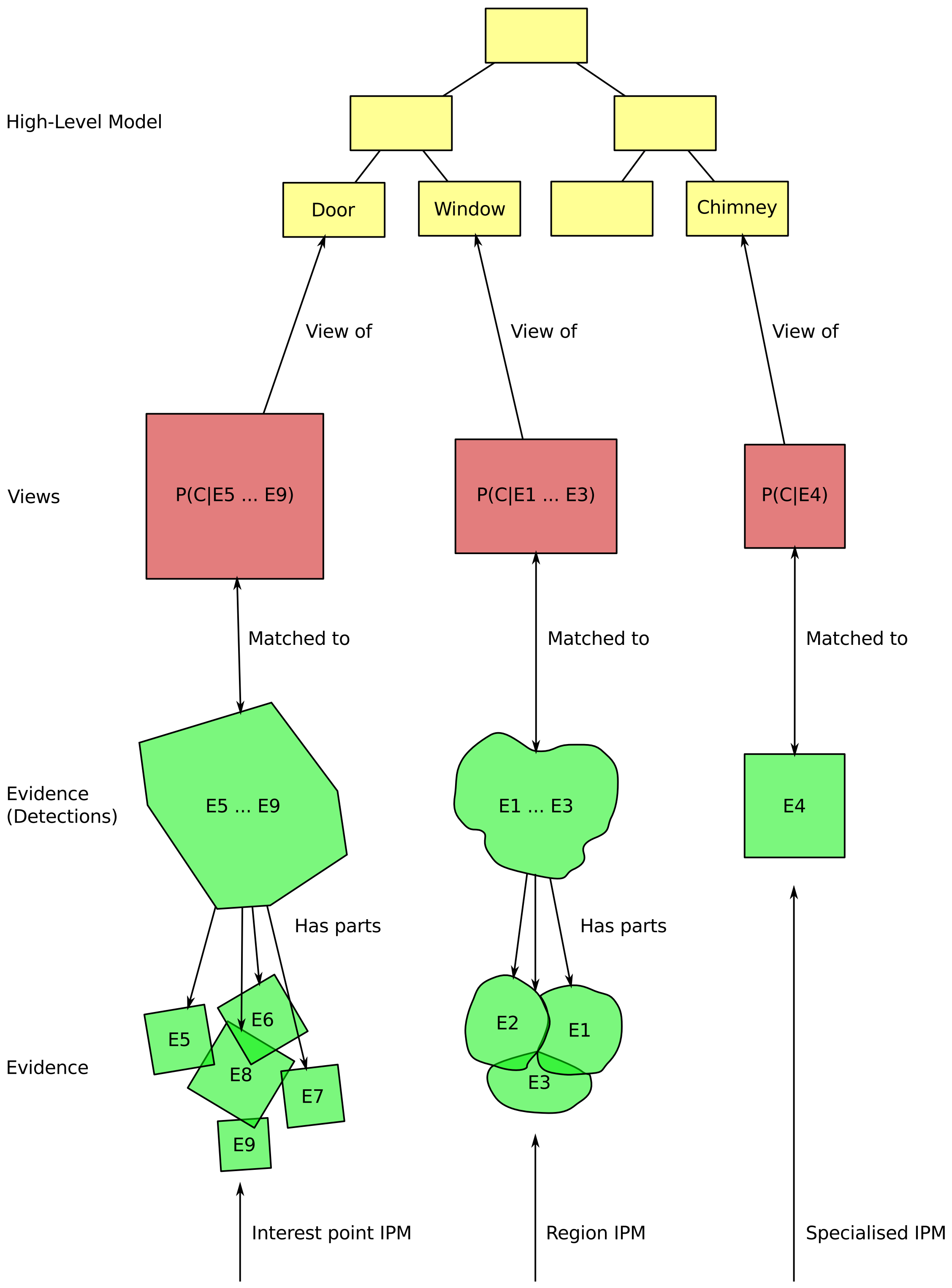


FIGURE 3.2: The middle-layer architecture. The high-level compositional model is shown at the top, the low-level evidence at the bottom. They are matched to each other through view concepts, shown in the middle.

### 3.1.2 Rationale for Incremental Interpretation

Since the space of all possible evidence assignments is huge in both probabilistic and crisp models, an exhaustive search for the best explanation of observed evidence is rarely feasible. At the same time, the high-level system and image processing modules can support each other during an interpretation process. Image processing modules can provide evidence which is integrated into the interpretation by setting some of the parameters of the model, thus reducing the complexity of the remaining assignments. Similarly, the integrated evidence represents scene context, which changes the expectations for further objects which can be observed. Such contextual information can help solve the difficult grouping and classification problems at the low level.

The proposed architecture approaches this problem using an iterative process. The interpretation is performed in a loop, with the evidence integrated in a stepwise fashion, and the high level and low level support each other after each iteration. Such an interpretation not only allows the interpretation process to start with easy assignments and use the context to resolve the more ambiguous decisions at a later point, but is also well suited for interpreting dynamic scenes, where evidence naturally comes in batches, and reasoning with incomplete data is expected.[1]

There are many ways to decide which evidence to integrate first. One consideration is to choose initial assignments which help reduce the remaining complexity by eliminating many potential explanations, such as detecting objects which are only possible in some models and whose role is not ambiguous. Another consideration is that some algorithms require significant processing power, and a coarse-to-fine strategy can make easy assignments first, and use the resulting context to focus the expensive algorithms on small portions of the image. Finally, poor initial assignments result in poor context and can degrade the overall performance, resulting in poor interpretations or leading to backtracking. The ideal integration sequence will depend on the individual interpretation system and the chosen domain, and must strike a compromise between combinatorial complexity, computational cost, and quality of the resulting interpretation. The integration strategy followed in this thesis was to always integrate the assignments with the highest confidence, regardless of the processing cost or complexity.

Stepwise matching of evidence to high-level concepts requires a classification methodology which can use the current scene context in order to detect and classify scene objects one by one. A probabilistic framework for stepwise classification is presented in the following section.

[1]Such a stepwise process represents a greedy search for the best interpretation, and is prone to compound errors based on wrong decisions. This problem can be tackled using backtracking or parallel interpretations.

## 3.2 Framework for Context-Based Classification

Since the evidence classification is done probabilistically, the starting point for developing a stepwise classification framework is a probabilistic interpretation. Let a set of vector-valued hidden variables $R_{all} = \{R_1 \ldots R_N\}$ describe the objects in a scene, where each object is an instance of a concept from a high-level domain ontology. Let $E_{all} = \{E_1 \ldots E_M\}$ be a set of vector-valued random variables describing the observed evidence in the scene, produced by the hidden variables. In Computer Vision, these can be regions, interest points, etc. A scenario modelling a certain kind of scene can be expressed as a large joint probability distribution $P(R_{all} E_{all})$. Finding the best interpretation for observed evidence $e_{all}$ can be seen a search for the global maximum in this JPD:

$$R_{all}^{solution} = \arg\max_{r_{all}} P(R_{all} = r_{all}, E_{all} = e_{all}) \tag{3.1}$$

There can be many alternative scenarios modelling different possible scenes in a domain, but this is ignored for the moment.

Typically, there is a huge number of possible ways to group the available evidence and assign it to the hidden variables from $R_{all}$, and the cardinality of $R_{all}$ and $E_{all}$ do not usually match. Often, a single object is described by more than one detected piece of evidence (for example regions or image patches). Also, some high-level objects can not be directly observed but are infered from other objects (such as complex activities, or arrangements of objects). In order to reduce the complexity, an intermediate set of variables is introduced, called *views*. The scenario is modelled as

$$P(R_{all} V_{all}), \tag{3.2}$$

where views $V_{all} = \{V_1 \ldots V_K\}$ are manifestations of real objects in the image. Each view $V$ is desribed by a feature vector grouping one or more discovered pieces of evidence, and additional information about the view: $V_k = \{bbox_k, position_k, E_{k1} \ldots E_{kP}\}$.

Assuming that each $V_k$ corresponds to one primitive object in the scene, and that they all have the same number of dimensions, this JPD can model arbitrary correlations between objects' position, size and appearance. The cost is the complexity of the resulting distribution: object appearance is complex to describe and feature vectors with hundreds of dimensions are common. Furthermore, modern object recognition and classification algorithms build appearance models from small object parts (image patches, superpixels, regions, etc.) and the resulting assignment problem (grouping the evidence together and mapping it to views) is extremely computationally complex.

This thesis proposes a simplification of the probability distribution by separating the evidence which is highly correlated and needs to be modelled in the JPD, from the evidence which is only loosely correlated between the objects in a scene, and is modelled using separate probability distributions. The JPD is a part of the high-level System, and the distributions modelling the local appearance of objects are a part of a separate middle layer. Both layers can provide a probabilistic classification of objects in the scene, based on the evidence available to them, and the middle layer combines the two classifications and instantiates views. The interpretation follows as an incremental process, with views being created one at a time, and the improved probabilistic context in the high-level system is used to improve the grouping and classification of remaining evidence.

Assume for the moment that the JPD $P(R_{all}V_{all})$ is available. Also assume that a view $V_k$ is represented only by a class $C_k$ and a (discrete) position $X_k$, making the JPD much easier to manage than if all evidence were directly modelled. The probability that there is a view of class $c$ in the image at a position $x$ can be obtained by setting $C$ and $X$ to $x$ and $c$ for each $V \in V_{all}$ in turn, marginalising over all the other variables, summing up over all $V_{all}$, and normalising the probabilities so they sum up to one.[2] This gives the joint probability $P(C = c, X = x)$. Dividing by $P(X = x)$ gives the conditional probability $P(C = c|X = x)$, which can be understood as a position-conditional prior for class $c$ given by the model, regardless of any observation. If $x$ is not a position, but a range of allowed positions given by a bounding box, then the value is the prior for class $c$ in a given area.

Now assume that a view $v_a = \{c_a, x_a\}$ has been integrated into the interpretation at a previous step and assigned to the hidden variable $r_a$. Given this information, the same process is performed using the distribution $P(R_1 \dots R_{N-1}, V_1 \dots V_{K-1}|r_a d_a)$. This gives the contextual posterior based on the previously integrated evidence. It can be seen that the JPDs provide a natural way to obtain class probabilities based on partial interpretations.

In the general case, there may be many possible scenarios modelling different scenes of a domain. Since each probability distribution $P(R_{all}V_{all})$ models the probabilistic constraints between a particular set of objects, the domain can be modelled as a combination of alternative scenarios. As presented in [Neumann and Terzić, 2010], the individual models are combined using a random variable $S$ with $s \in 1 \dots M$ identifying different scenarios and the probability $P_S(S)$ giving the probability of a given scenario:

$$P_{scenario} = P_S(S)P^S(R^S_{all}, V^S_{all}) \tag{3.3}$$

The class priors then have to be summed up across all alternative models and weighted by the prior probabilities of the models. The following sections describe how such contextual information can be used to improve classification.

[2]The normalisation step is necessary because there can be numerous objects of the same class in the scene, and different objects might overlap. It should be noted that the normalisation is only correct under the assumption that the object is actually present in the scene (no modelling of false positives in the joint probability). Dealing with false positives is handled elsewhere, as described in Section 3.3.4.1.

### 3.2.1 Independence of Structural and Appearance Information

Two assumptions are made in order to provide a mechanism for classifying evidence in a stepwise interpretation process. Assume for the moment that the appearance of an object in an image is described by a complete feature vector $E_C$, which represents the collected evidence for one object in the scene, provided by an image processing module. The first assumption is that the evidence vector $E_C$ can be split into an appearance part and a structural part:

$$E_C = \left\{ \frac{E_s}{E_a} \right\}, \tag{3.4}$$

where $E_a$ and $E_s$ are not correlated if it is known which view produced them, so

$$P(E_C|C) = P(E_a|C)P(E_s|C). \tag{3.5}$$

If $E_s$ represents the object location, and is therefore dependent on the location of other objects in the scene, it can be used to reason about the positions and classes of the objects in the scene. $E_a$ models the appearance of the object regardless of its position in the image, such as colour, shape, spatial arrangement of parts, etc. Since the structural information $E_s$ is a part of the view description and does not depend on $E_a$, the remaining task is to determine the class of the view, and the likelihood term $P(E_a|V)$ can be written as $P(E_a|C)$, as is common in Computer Vision.

The second assumption is that the typical appearance of the classes is not affected by context. During incremental interpretation, the understanding of the scene is based on already matched evidence, and each new classified view affects the current understanding of the scene and thus the prior probabilities of remaining objects. It is assumed that

$$P(E_a|C) = P'(E_a|C), \tag{3.6}$$

where $P'(E_a|C)$ is the evidence likelihood after a change in scene context, e.g. after some evidence has been matched. The main idea is to use local appearance to detect and classify views, and not the absolute position in the image. This assumption is common in object detection.

An obvious consequence of these independence assumptions is that appearance correlation between scene objects is ignored. This seems like a rather harsh restriction at first. For example, it seems intuitive that all windows in a façade should exhibit strong similarity. In this case, the simplifying assumption about the independence of appearance from context can mean losing performance. However, there is a large number of domains where scene interpretation is used in which this does not play a role. For example, crowd monitoring scenarios cannot expect the visual appearance (height, clothes or similar) of the individuals to be correlated. Similarly, traffic interpretation systems deal with a large number of visually dissimilar vehicles, and knowing the appearance of one car will generally not aid in classifying another.

In the case where correlations between objects of certain classes in the image play a significant role (e.g. the colour or shape of one object will directly affect the colour and shape of the other objects), one can reformulate the problem by splitting the problematic class into several distinct classes. For example, in the façade domain, one might introduce a "T-style window" class and a "cross-style window" class to the high-level ontology and learn the appearance separately. This way, correlations between objects can be modelled at the high level (by letting all windows in a horizontal row have the same sub-class), and the independence assumption holds. As with many modelling problems, there is a trade-off between the complexity of the model (adding additional classes) and the improvement that the more complex model brings.

### 3.2.2 Integrating Context Prior Using Evidence Likelihood

Assume that a view $v$ delivered by an image processing module is described by an evidence vector $e_C = [e_s, e_a]^T$ and needs to be classified. It is assumed that $E_s$ and $E_a$ are independent given C. Also, $E_a$ does not change depending on the structural context, so it is independent of the previously integrated evidence $\underline{E}_{k-1}$ at the interpretation step $k$. Therefore, the classification equation for a single object during interpretation step $k$ can be derived as follows:

$$P(E_C C) = P(E_a E_s C) = P(E_a|C)P(E_s C) \tag{3.7}$$

$$P(E_C C|\underline{E}_{k-1}) = P(E_a|C\underline{E}_{k-1})P(E_s C|\underline{E}_{k-1}) = P(E_a|C)P(E_s C|\underline{E}_{k-1}) \tag{3.8}$$

Classification of $E$ given previous assignment of $\underline{E}_{k-1}$:

$$P(C|E_a E_s \underline{E}_{k-1}) = \frac{P(CE_a E_s|\underline{E}_{k-1})}{P(E_a E_s|\underline{E}_{k-1})} = \frac{P(E_a|CE_s\underline{E}_{k-1})P(CE_s|\underline{E}_{k-1})}{P(E_a E_s|\underline{E}_{k-1})} \tag{3.9}$$

Due to the independence assumption, $P(E_a|CE_s\underline{E}_{k-1}) = P(E_a|C)$, giving:

$$\frac{P(E_a|C)P(CE_s|\underline{E}_{k-1})}{P(E_a E_s|\underline{E}_{k-1})} = \frac{P(E_a|C)P(C|E_s\underline{E}_{k-1})\cancel{P(E_s|\underline{E}_{k-1})}}{P(E_a|E_s\underline{E}_{k-1})\cancel{P(E_s|\underline{E}_{k-1})}} \tag{3.10}$$

$$= \frac{P(E_a|C)P(C|E_s\underline{E}_{k-1})}{P(E_a|E_s\underline{E}_{k-1})} \tag{3.11}$$

$$= \frac{P(E_a|C)P(C|E_s\underline{E}_{k-1})}{\sum_C P(E_a|C)P(C|E_s\underline{E}_{k-1})} \tag{3.12}$$

$$P(C|E_C, \text{context}) = \frac{P(E_a|C)P(C|E_s, \underline{E}_{k-1})}{P(E_\text{a}|\underline{E}_{k-1})} \tag{3.13}$$

This provides the central classification equation for stepwise probabilistic scene interpretation:

$$P(C|E_C, \text{context}) = \frac{\overbrace{P(E_a|C)}^{\text{appearance model}} \quad \overbrace{P(C|E_s, \underline{E}_{k-1})}^{\text{contextual prior}}}{\underbrace{P(E_\text{a}|\underline{E}_{k-1})}_{\text{normalisation}}} \tag{3.14}$$

Since we assumed that the typical appearance of the classes is not affected by context (Equation. 3.3.3.1), the probability $P(E_a|C)$ that an evidence described by a feature vector $E_a$ was generated by an object of class $C$ can be written as

$$P'(C|E_a) = \frac{P(E_a|C)P'(C)}{P'(E_a)} = \frac{P(E_a|C)P'(C)}{\sum_c P(E_a|C)P'(C)} \tag{3.15}$$

The formulation in Equation 3.14 allows for introducing updated priors $P'(C)$, which reflect the scene context coming from incremental high-level intepretation or an additional knowledge source. This presents a generic way to introduce scene context into the classification process. If an updated prior $P'(C)$ is not available (at the beginning of an interpretation before any context is known, or when using a crisp high-level system), a domain prior can be used instead, evaluated on a representative set of images.

Any classifier which provides $P(E_a|C)$ can be modified in this fashion, as long as the evidence vector $E_a$ does not include any information from $E_s$ which was used to calculate the updated priors (Equation 3.5). This is a very common situation in scene interpretation, where object detection algorithms generally act on local appearance, and interpretation models the scene in terms of absolute and relative positions of the scene objects. In the rest of this chapter, it is assumed that the evidence used by the low-level appearance model $P(E|C)$ does not include any structural information, and $E$ is used to mean appearance information only.

### 3.2.3 Update Mechanism for Posterior Class Probability

Estimating the likelihood $P(E|C)$ is difficult if $E$ is high-dimensional, so classification schemes typically estimate $P(C|E)$ and not $P(E|C)$. Under the assumption made in Equation 3.3.3.1, an update mechanism can be derived to calculate $P'(C|E)$ from $P(C|E)$, $P(C)$ and $P'(C)$.

$$P'(C|E) = \frac{P'(CE)}{P'(E)} = \frac{P'(E|C)P'(C)}{P'(E)} = \frac{P(E|C)P'(C)}{P'(E)}$$

$$= \frac{P(EC)\frac{P'(C)}{P(C)}}{P'(E)} = \frac{P(C|E)P(E)\frac{P'(C)}{P(C)}}{P'(E)} \tag{3.16}$$

$P'(E)$ can be expressed as

$$P'(E) = \sum_c P'(CE) = \sum_c P(E|C)P'(C)$$

$$= \sum_c P(EC)\frac{P'(C)}{P(C)} = \sum_c P(C|E)P(E)\frac{P'(C)}{P(C)}$$

$$= P(E)\sum_c P(C|E)\frac{P'(C)}{P(C)} \tag{3.17}$$

Inserting 3.17 into 3.16 gives

$$P'(C|E) = \frac{P(C|E)\frac{P'(C)}{P(C)}}{\sum_c P(C|E)\frac{P'(C)}{P(C)}} \tag{3.18}$$

where $P(C)$ are the domain priors of the training set used for learning the probabilistic classifier, $P(C|E)$ are the conditional class probabilities, and $P'(C)$ are the updated class priors. This formulation allows easy integration of context priors, provided a good estimate of $P(C|E)$ is available.

### 3.2.4 Classification vs. Role Assignment

The term "classification" has been used in this thesis to mean assigning a class from an ontology to an observation. In a realistic interpretation scenario, ontologies used to describe the domain are often large and based on semantic as well as visual concepts. Figure 3.3 illustrates this using a typical example from the façade domain. The "Window-Array" concept describes an arrangement of windows, and as such can not be directly observed, only inferred from the observed windows. The three windows belonging to the window array all look identical, but represent different concepts. Matching the window evidence to one of these concepts is a type of classification, but it can not be done based on the appearance alone, since they all three windows have virtually identical appearance. Learning a specialised "Left-Window" detector makes little sense and higher-level context about the relative positions of windows is needed to make the

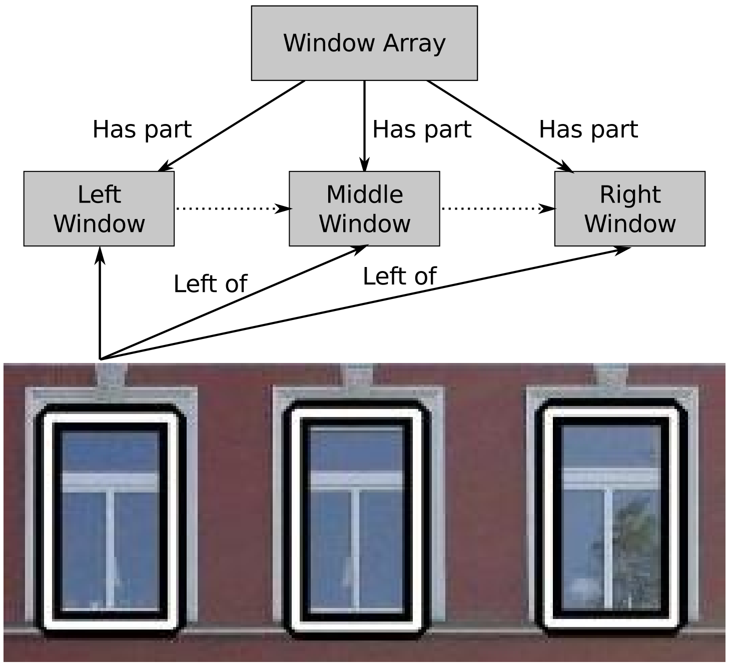


FIGURE 3.3: The role assignment problem. After some evidence is classified as a window, there is still ambiguity about its role in the high-level model. There are six possible ways to assign the three windows to the three high-level objects, but only one set of assignments satisfies the spatial constraints between the windows. In the diagram above, the solid lines between objects represent part-of relations, the dashed lines represent constraints between objects and the arrows at the bottom represent possible assignments for the leftmost detection.

correct assignment. This type of problem is referred to as **role assignment**. In a typical image from the façade domain, there are many possible roles for every window, like “leftmost window in the second floor window array”, “right balcony window in the left balcony on the third floor”, etc. The role assignment quickly becomes one of the main causes of complexity.

There are two ways to deal with the role assignment problem. The first one is to classify the detections based on a subset of the complete ontology. In the example shown in Figure 3.3, this would mean creating a “Window” view, since “Window” is a more general concept than “Left-Window” or “Right-Window”. The High-level system then has to find the best assignment for this view by considering the position of the window and the relation to other objects, and specialising the concept to one of the more specific classes. This approach essentially leaves the role assignment to the high-level System.

The second approach is to use the probabilistic priors from the high-level system. The priors are delivered for classes “Left-Window”, “Right-Window” and “Middle-Window” separately. If $P(C|E)$ is not available for these classes, the middle layer assumes that their appearance is the same, and uses $P(C|E)$ for the closest more general concept, in this case “Window”. The classification is then performed as described Eq. 3.18, using $P'(\text{Left-Window})$ and $P(\text{Left-Window}|E) = P(\text{Window}|E)$.

### 3.2.5 Classification for Crisp Interpretation

The equations in these sections were developed from a probabilistic view of scene interpretation. The separation of the problem into view classification and role assignment means that probabilistic classification of views does not require a probabilistic high-level model, as long as the views are provided to the high-level as already classified (or a list of possible classifications in the general case). The crisp interpretation systems can accept such classified views and apply relevant rules after integrating them, regardless of how the classification was performed.

In this case, a contextual prior $P'(C)$ is not available and has to be replaced by a static domain prior. This is still a useful property for generic image interpretation. Some object detection algorithms (such as the one presented in Section 3.3.3) make assumptions about the class priors, typically assuming that all classes are equally likely. Such models can be learned using a training set which fulfils these requiements, and used for scene interpretation in a more realistic setting, by updating the probabilities produced by such a detector to reflect the static domain priors.

Furthermore, the use of probabilistic classifiers at this stage means that a measure of confidence is available for each classification decision. The middle layer will typically integrate the evidence with high classification confidence first (see Section 3.5), leaving the more difficult decisions for later, when high-level hypotheses can help to disambiguate the evidence based on high-level context, as described in Section 3.6. Since the interpretation is generally obtained as a stepwise process, starting with most certain evidence assignments reduces the chance of propagating the wrong context early, which can lead to poor interpretations.

Finally, representing the classification as a probability distribution $P(C|E)$ acts as a way to model the classification uncertainty. Since it is evaluated on detections for each detector separately, the classification probability also represents the confidence in the detection. Detectors which are very accurate at detecting certain classes of objects will have higher classification probabilities for those classes than detectors which deliver many false positives. This provides a way to compare the results of several detectors.

The following section describes the process of obtaining probabilistic descriptions for a wide range of low-level evidence.

## 3.3 Low-Level Integration

Low-level evidence can be obtained from an image by a multitude of computer vision algorithms. They differ in terms of precision (from blob-like detectors to sub-pixel operators), scale (from edgels and interest points to regions encompassing complete objects) and specialisation (from general segmentations to detectors trained to detect a specific class of objects).

One of the goals of generic middle-level image processing is to provide a unified interface that allows an interpretation system to use the best tools available. As explained earlier, this requires modelling the classification probabilities and size and position uncertainties of each individual detector. In terms of probability, this amounts to estimating $P(C|E)$ given an evidence feature vector $E$ provided by the low level. $P(C|E)$ can take one of three forms, dividing the detectors into three general types:

- specialised classifying detectors, where $E$ is a label representing a class from some ontology,
- non-classifying segmenters, where $E$ is a feature vector describing the properties of the region,
- structured models built from parts, where $E$ is decomposed into a number of smaller feature vectors describing the individual patches and the spatial relationships between them.

**Specialised classifying detectors** detect instances of objects of a certain class in an image. They have been used for a wide range of applications, generally when the object of interest has a predictable appearance that can be modelled well. Popular examples include the detection of faces [Viola and Jones, 2004], pedestrians [Leibe et al., 2005] and license plates [Zhang et al., 2006]. Though they use complex models internally, they are often used as a black box which provides labelled regions with the labels corresponding to an ontology. In many cases, they do not provide a calibrated class probability, which means it has to be estimated in order to integrate such detections into a probabilistic framework. This means that the error rate of the classifier must be evaluated on an annotated dataset. As $E$ is a symbol in this case (the class provided by the detector), $P(C|E)$ has the form of a table, showing the classification confidence for each possible value of $E$.

**Non-classifying segmenters** detect regions which roughly correspond to objects. In general, detecting object contours in a bottom-up fashion is difficult. However, there are some special cases in which the objects can be detected well because they have a specific colour (such as skin detection) or shape (rectangular, elliptic or rotationally symmetric objects), or because an additional source of information can be used for segmentation (such as a thermal image or a laser range scanner). The most common source of unclassified object detections in images is tracking in video sequences, which typically produces pixel masks which have to be classified based on image content within the identified region (see Figure 3.5 for an example). In this case, the detected region first has to be described by a feature vector $E$, and the class probability $P(C|E)$ must be estimated in a separate step for each class and each possible value of $E$.

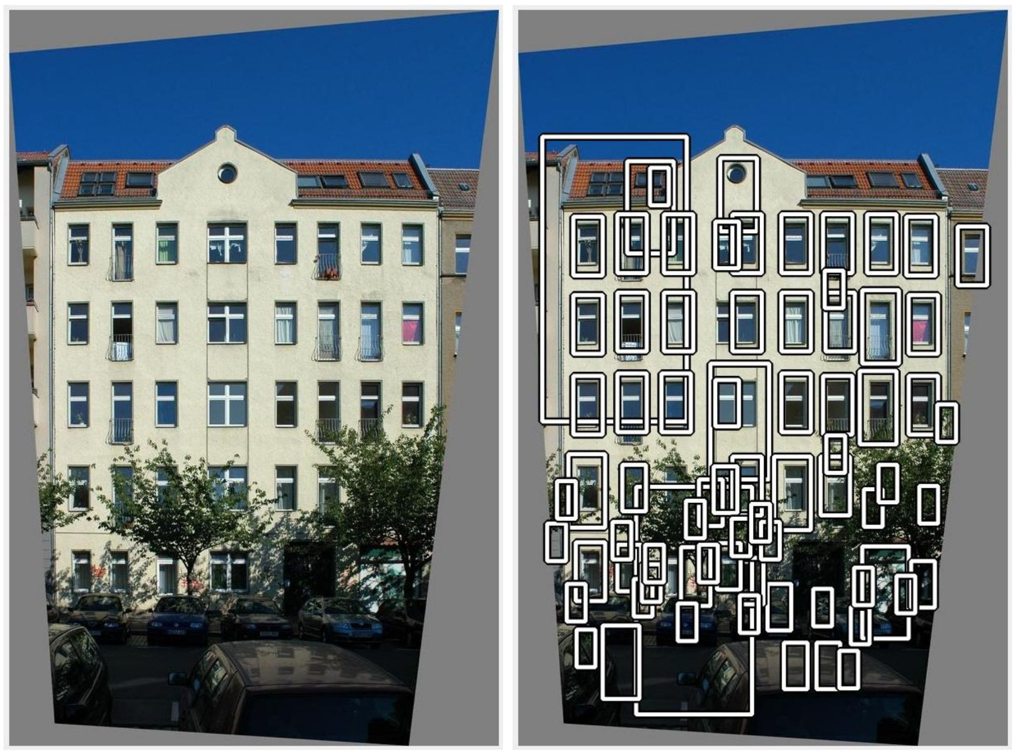


FIGURE 3.4: The output of a specialised detector trained to detect windows at different scales. Although many windows are found, there are also many false positives, and some doors belonging to French balconies are identified as windows. Due to these errors, the results cannot be taken at face value and the error rate must be estimated by comparing the detections to an annotation. The detector used is based on Adaboost and trained to recognise windows, as in [Terzić et al., 2010].

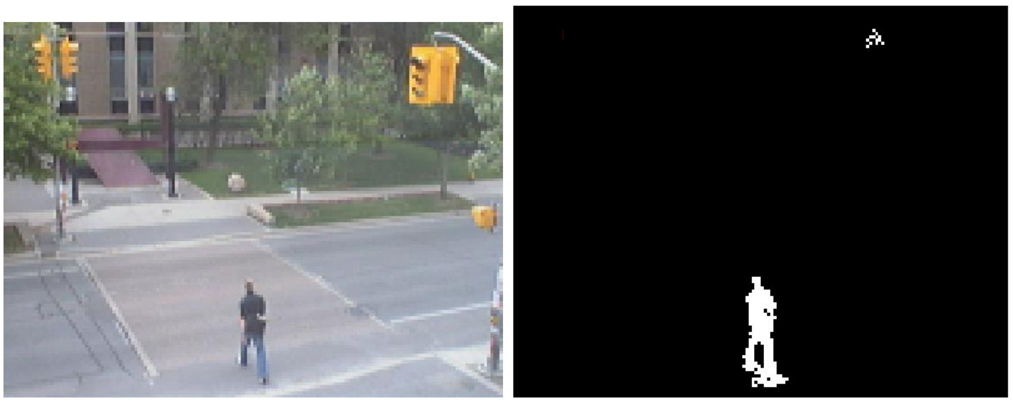


FIGURE 3.5: Output from a tracker showing an outline of a person. The region is detected because of motion across several frames, but the region is not classified. Reproduced from *http://www.cs.berkeley.edu/~flw/tracker/*.

**Structured models built from parts** detect salient image patches corresponding to object parts, either by using interest point detectors like [Ommer and Buhmann, 2010, Leibe and Schiele, 2003, Fergus et al., 2007], or by using interesting regions produced by an oversegmentation [Gu et al., 2009]. Each part contributes to the overall segmentation, and the relation of the parts with respect to the object centroid and each other is taken into account. In this case, $E$ is decomposed into a number of smaller vectors modelling individual parts and their relations to each other, and the probability distribution $P(C|E)$ is put together, usually with the help of some independence assumptions. In recent years, structured patch-based models have been very successful at modelling and detection of a wide range of object categories and dominate the results of popular benchmarks such as Caltech 101 [Fergus and Perona, 2003] and Visual Object Challenge [Everingham, 2006]. In addition to the state-of-the-art detection performance, these models are appealing because they model objects as a combination of simpler parts, just like complex objects are modelled as a combination of simpler objects in the high-level of an interpretation system. This means that such structured models represent a natural extension of a high-level hierarchy down to the level of image patches.

### 3.3.1 Classification Confidence for Specialised Detectors

Specialised detectors attempt to detect instances of objects in an image directly. In the context of this section, this encompasses all detectors which provide a classification together with a region, even if they can detect and classify objects of several classes. Classic scene interpretation schemes like [Mohnhaupt and Neumann, 1993, Fusier et al., 2007, Nagel, 2004] often use such a bottom-up driven approach. It can work well for certain classes of objects or restricted domains like faces or cars, but is also error-prone in the general case for two main reasons. First, the object is classified based on the appearance alone, without the use of higher-level contextual knowledge. In many domains, there is a significant overlap between the appearance of different object classes, leading to misclassifications. Second, any detection error is propagated into the high-level, leading to wrong interpretations, or violating the scene model. Because of these reasons, it is important to model the error of any specialised detector, as a measure of how much the detections can be trusted.

Figure 3.4 shows an example of an AdaBoost-based classifier of [Šochman and Matas, 2009] trained to detect instances of T-style windows. It returns regions labelled as "window", representing a potential window detection. It can be seen that it detects many windows in a typical scene, but also that there are many false positives. This is not surprising, as the window class is tricky to model due to a huge variation in appearance [Terzić and Neumann, 2009a] within the class. It can also be seen that some doors are classified as windows. This is also not surprising, since doors and windows often have similar appearance, and considering that the detected instances are doors of French balconies, which are often understood to be windows by humans, due to a semantic ambiguity about what constitutes a door. Still, this example clearly shows that an interpretation system cannot accept such evidence as fact, and that modelling the classification confidence is needed.

Probabilistically speaking, the classification uncertainty for evidence can be expressed as a probability $P(C|E)$, where $E \in \{K_1, K_2, \ldots, K_M\}$, and $K_m$ are symbols representing all the classes used to train the detector. In the façade example, a detector can be trained to detect windows (and deliver detections with a "window" label), but such detections will often include doors and false positives, and the $P(C|E)$ term expresses the classification confidence for each class in $C$.

The classification probability $P(C|E)$ in this case can be evaluated on an annotated set using a frequentist approach. For each detection with a sufficient overlap with an annotated object (e.g. using the PASCAL criterion), the detected class and the ground-truth class are noted, giving $P(C_n|K_m)$ for all $n \in \{1 \ldots N\}$ and $m \in \{1 \ldots M\}$. Since both $C$ and $E$ are discrete, the probabilities can be efficiently stored in a lookup table. The resulting $P(C|E)$ can be plugged into Equation 3.18 to add contextual priors and help disambiguate (e.g. between doors and windows). $P(C)$ and $P(E)$ can also be obtained from the annotated set in the same way.

If the IPM provides a probabilistic confidence rating with each detection, this can be used directly as a value for $P(C|E)$, without having to evaluate the detections using ground truth. The experiments presented in this thesis have not made use of the IPM confidence rating, and have evaluated the performance on an evaluation set to estimate $P(C|E)$.

### 3.3.2 Estimating Global Appearance Model for Non-Classified Regions

In the case where the outline of the object is known (provided by a segmenter or a tracker), but the detection is not classified, it is neccesary to estimate the conditional probability $P(C|E)$, where $E$ is a feature vector describing the region and $C$ is one of the classes from the high-level ontology. The probabilistic context can then be integrated by using Equation 3.18. More formally, given a region $R$ described by an $M$-dimensional feature vector $E = \{E_1, E_2, \ldots, E_M\}$, where $E_m \in \mathbb{R}$ are unary measurements, the task is to estimate the class posterior $P(C|E)$ for each $C \in \{C_1, C_2, \ldots, C_N\}$.

This probability can be estimated by a probabilistic multi-class classifier, such as Support Vector Machines (SVM) or Non-linear Kernel Discriminant Analysis (NKDA), or automatically learned decision trees, as described in [Terzić and Neumann, 2009a]. In this section, two common classifiers used in this framework are presented.

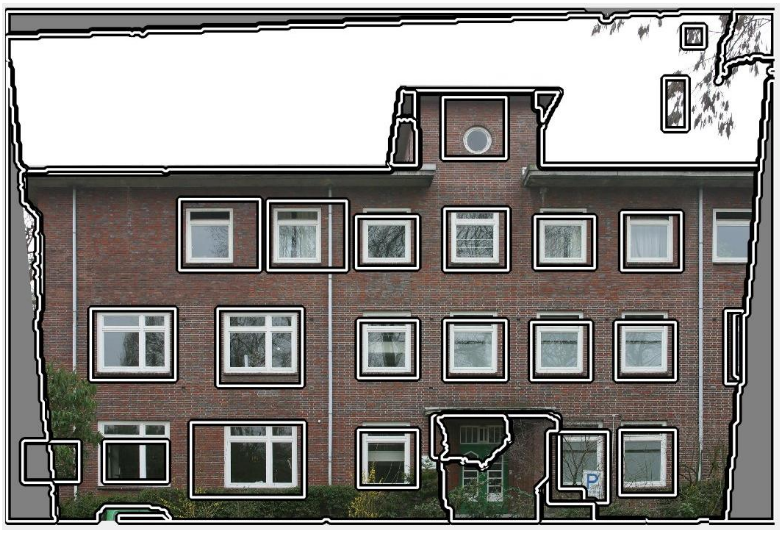


FIGURE 3.6: Output from the interest region detector of [Jahangiri and Petrou, 2008]. The interesting regions are detected using bottom-up saliency measures and not object models. It can be seen that the region detector produces spurious detections caused by vegetation, and that the detections are not precise. Each detected region is described by a feature vector.

#### 3.3.2.1 Classification using Decision Trees

A decision tree is a tree where the leaf nodes represent classifications and each non-leaf node represents a decision rule acting on an attribute of the input sample. A sample described by a $d$-dimensional feature vector $f$ and consisting of $d$ scalar attributes is classified by evaluating the decision rule at the root node and passing the sample down to one of the subnodes depending on the result, until a leaf node is reached. The result is a partitioning of the feature space into labelled disjoint regions.

If the leaves of a decision tree are allowed to correspond to more than one class, they are called impure leaves. If each class in a leaf node is given a probability, such trees can be used to model the uncertainty of the classification result. Essentially, they provide a discrete approximation of a probability density function, where the discretisation can be irregular.

If an impure leaf $l$ contains the samples of several classes, an estimate of $P(C|L)$ for all classes and leaves can be formulated as

$$P(c|l) = \frac{N_c(l)}{N(l)}$$

where $N_c(l)$ is the number of samples in $l$ belonging to class $c$ and $N(l)$ is the number of all samples in $l$. The probabilities at the leaves $P(C|L)$ reflect the success rate achieved with the training set used to learn the tree.

Instead of encoding $P(C|L)$, we can observe how often an object belonging to class $c$ generates evidence described by the leaf $l$, giving the class-conditional probability

$P(L|C)$ for all classes and leaves. Then, Bayes rule gives the posterior probability as

$$P(C|L) = \frac{P(L|C)P(C)}{P(L)} \tag{3.19}$$

Finding the class for which $P(C|L)$ is maximum gives a MAP classifier. Since each evidence sample $e$ is mapped into a leaf $l$ of the decision tree, $P(C|L)$ serves as a discrete approximation of $P(C|E)$.

For many problems, decision trees have competitive performance compared to other classification schemes [Webb, 2002]. At the same time, they have the advantage of having a result that can be understood intuitively because they split the feature space into regions with axis-parallel boundaries. In addition to providing bottom-up classification of low-level evidence, they can also *describe* the visual appearance of high-level concepts by specifying a region of feature space. This is an appealing property for scene interpretation, because it simplifies top-down processing by making it possible to pass expectations of low-level appearance of expected objects to image processing algorithms in a compact and understandable form. A more detailed description of decision tree learning is available in Appendix A.

In our experiments, decision trees performed very well in low-dimensional feature spaces (up to 18 dimensions) without requiring any parameter tuning, which makes them very appealing. In higher-dimensional spaces (upward of 200 dimensions), which are necessary for more complex structured appearance models, SVMs performed far better due to the use of nonlinear kernels.

#### 3.3.2.2 Multi-Class Support Vector Machines

The probabilities needed in the middle layer can also be estimated using Support Vector Machines (SVMs). This section outlines the challenges involved.

SVMs [Vapnik, 1995] are binary classifiers which map the feature vectors into a high-dimensional feature space and attempt to find the best separating hyperplane. Several approaches exist for solving multi-class problems (like the ones encountered in Computer Vision) using SVMs. A multi-class SVM formulation exists [Vapnik, 1998], where $N$ linear discriminant functions are constructed simultaneously for $N$ classes. The problem with this approach is that the addition of constraints for each class simultaneously results in a quadratic optimisation problem which is difficult to solve [Crammer et al., 2001]. Although Crammer and Singer proposed a more efficient solution, which was implemented and is available as software [Joachims, 1999], the multi-class solution tends to be complex and perform worse than a combination of binary classifiers both in our experiments and in other reserchers' surveys [Duan and Keerthi, 2005].

Multi-class problems are generally formulated as a combination of $N$ one-against-all binary classifiers, or $N(N-1)/2$ one-against-one binary classifiers. The one-against-all classifier tests each class against the samples from all other classes and gets a confidence rating indicating how far the sample is from the hyperplane separating the class from all others. The class with the highest response wins. The one-against-one classifier performs $N(N-1)/2$ pairwise classification tasks between each possible pair of classes, each giving a vote. The class with the highest number of votes is the winner.

More sophisticated methods based on the one-against-one approach were proposed, such as *pairwise coupling* [Hastie and Tibshirani, 1998] and Directed Acyclic Graph SVMs [Platt et al., 2000], which reduce training time, but a recent survey [Hsu and Lin, 2002] showed their performance to be comparable to that of standard one-against-one classifiers.

Although SVMs are known to perform well on a wide range of classification problems, they do not produce probabilities but a value expressing the distance of the sample to the hyperplane. Since this value is uncalibrated, care must be taken when constructing multi-class classifiers from a set of binary ones, and all samples must be normalised the same way. There are several methods for obtaining probabilities from these uncalibrated values, such as parametrising the distance in the dimension perpendicular to the hyperplane and expressing the likelihood as a weighted sum of cosines [Vapnik, 1998], by using a logistic link function [Wahba, 1999], or by applying a sigmoid function to the SVM output [Platt, 1999]. The SVM library used for probabilistic classification in this thesis [Chang and Lin, 2001] uses pairwise coupling to estimate the probabilities internally [Wu et al., 2004].

The SVM work described in this thesis was done with one-against-all and one-against-one classifiers, and the probability estimates were calculated by the libSVM library using pairwise coupling. However, other methods, and other types of classifiers can also be used.

### 3.3.3 Structured Appearance Models

In recent years, there has been an increasing number of interest-point based approaches to object detection and classification in the Computer Vision community. The work presented in [Ommer and Buhmann, 2010] provides a probabilistic hierarchical approach to object classification, which is relevant for scene interpretation for several reasons:

- the object structure is learnt and not hand-modelled, which makes it applicable to domains with hundreds of classes,
- it is probabilistic, making it a good fit for probabilistic context-based classification, and

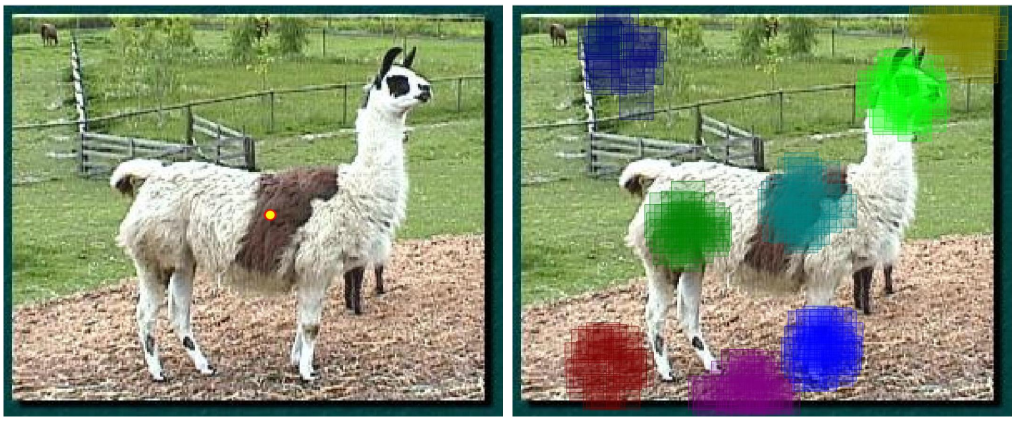


FIGURE 3.7: Compositions of atomic parts shown in an example. Left: an image showing a llama, with the object centre indicated by a red circle. Right: eight randomly sampled "compositions" $g_j$ based on a proximity criterion. Each composition consists of a number of rectangular image patches centered around interest points detected by a Harris detector. The relevant compositions (those on the llama) are distinguished from the irrelevant ones (showing background) by a relevance function.

- it requires minimal annotation for learning.

In this section, the integration of this probabilistic visual model into the middle layer framework is presented.

Unlike many categorisation problems, where the task is to give a label to a whole image, scene interpretation attempts to find (and explain) all objects in the scene. The number of objects of any given class in the image is generally not known. The authors present a way to detect multiple objects and a method for finding the object centres, under the assumption of maximum one object per class in the image. Naturally, this assumption is too strong for the general case.

#### 3.3.3.1 Probabilistic Classification Based on Object Parts

In their work, Ommer and Buhmann characterise each feature point (found using a Harris-Laplace operator) by a 40-dimensional histogram of oriented gradients and colour. Each feature point can be characterised by a Gibbsian distribution over a codebook $\mathcal{V}$ of $k$ entries (they suggest $k = 200$):

$$P(F_i = \nu | e_i) = \frac{exp(-d_\nu(e_i))}{\sum\limits_{1 \le \nu \le k} exp(-d_\nu(e_i))} \tag{3.20}$$

Feature points are randomly grouped based on proximity, giving *compositions*. Each composition $\Gamma_j$ is described by a $k$-dimensional descriptor $g_j$

$$g_j = \sum_{e_i \in \Gamma_j} \frac{1}{|\Gamma_j|} \left(P(F_i = 1|e_i), \dots, P(F_i = k|e_i)\right)^T \tag{3.21}$$

All descriptors are normalised to zero mean and unit variance using the means and standard deviations derived from the training set, giving a representation which is robust with respect to the number of parts.

A visual context term $g^I$ describing the image $I$ is calculated as the average of all the $g_j$ drawn from the image. Furthermore, higher-order compositions (compositions of compositions) are randomly sampled from the compositions, giving tuples $(g_k, g_l)$ and a relation $r_{kl}$ between them (e.g. 2D distance vector). A set of relevant compositions $\mathcal{R}$ and relevant compositions of compositions, $\tilde{\mathcal{R}}$ is automatically selected from the randomble drawn composition candidates. Then, the classification problem is formulated as:

$$P(c|E) = \frac{p(\{g_j, s_j\}_{g_j \in \mathcal{R}}, \{g_k, g_l, r_{kl}\}_{(g_k, g_l) \in \tilde{\mathcal{R}}}, g^I | x, c) P(c|x)}{p(\{g_j, x_j\}_{g_j \in \mathcal{R}}, \{g_k, g_l, r_{kl}\}_{(g_k, g_l) \in \tilde{\mathcal{R}}}, g^I | x)} \tag{3.22}$$

$$P(c|E) = \frac{\prod_{g_j \in \mathcal{R}} p(g_j, s_j|x, c) \times \prod_{(g_k, g_l) \in \tilde{\mathcal{R}}} p(g_k, g_l, r_{kl}|x, c) \times p(g^I|x, c) P(c|x)}{\prod_{g_j \in \mathcal{R}} p(g_j, s_j|x) \times \prod_{(g_k, g_l) \in \tilde{\mathcal{R}}} p(g_k, g_l, r_{kl}|x) \times p(g^I|x)} \tag{3.23}$$

A density estimation for the needed terms is very difficult, so the following assumptions are made:

- There is one object in the image,
- the object class to be recognised is $c \in \mathcal{L}$, where $\mathcal{L}$ is the set of classes used during learning,
- the object position $x$ is known, and
- all classes a priori equally likely

In this case, the problem can be formulated as Maximum Likelihood classification:

$$P^*_{\mathcal{L}}(c|E) = exp\left[lnP(c|g^I) + \sum_{g_j \in \mathcal{R}} lnP(c|g_j, s_j) + \sum_{(g_k, g_l) \in \tilde{\mathcal{R}}} lnP(c|g_k, g_l, r_{kl})\right] \tag{3.24}$$

In this section, the assumptions made by the authors are relaxed in order to make it applicable for the general case of detecting multiple objects in cluttered scenes.

The $P^*_{\mathcal{L}}(c|E)$ term used for Maximum Likelihood classification is not a real probability. Although it is based on a closed-world assumption (all classes are known), the probabilities for the individual classes do not sum up to one, because it lacks the normalisation factor. In order to express the probability that the object at a certain location belongs to a certain class, the normalisation factor $F$ must be factored in:

$$P_{\mathcal{L}}(c|E) = P^*_{\mathcal{L}}(c|E) \times F, \tag{3.25}$$

where

$$F = \frac{P(c|x)}{\prod\limits_{g_j \in \mathcal{R}} p(g_j, s_j|x) \times \prod\limits_{(g_k,g_l)\in\tilde{\mathcal{R}}} p(g_k, g_l, r_{kl}|x)} \times \prod_{g_j\in\mathcal{R}} \frac{p(g_j, s_j|x)}{P(c|x)} \times \prod_{(g_k,g_l)\in\tilde{\mathcal{R}}} \frac{p(g_k, g_l, r_{kl}|x)}{P(c|x)}.$$

Due to the assumptions that the class is not affected by absolute position and that all classes are equally probable, the probability $P(c|x)$ is the same for all classes, so the factor $F$ is a class-independent constant. Calculating $F$ requires estimating all the probabilities from Eq. 3.3.3.1 in high-dimensional space, and this estimation was avoided by formulating the task as a Maximum Likelihood problem. However, due to the Assumptions 1 and 2, all class probabilities must sum up to one, giving:

$$\sum_{c\in\mathcal{L}} P_{\mathcal{L}}(c|E) = 1 \tag{3.26}$$

$$F = \frac{1}{\sum\limits_{c\in\mathcal{L}} P^*_{\mathcal{L}}(c|E)} \tag{3.27}$$

This trick makes it possible to have normalised probabilities, as long as Assumption 4 holds.

Since this formulation uses conditional class probabilities, Eq. 3.14 can not be used directly, but if the assumption presented in Eq. holds, the update mechanism presented in Eq. 3.18 can be used to update the class probabilities using new priors. This means that the classes only have to be equally likely during learning, and that more realistic priors (and even contextual priors) can be used during interpretation, thus removing the need for the 4th Assumption.

#### 3.3.3.2 Bottom-Up Object Detection

The remaining Assumptions 1-3 make this a pure classification (as opposed to detection) task. The classification framework can be used to verify high-level hypotheses which have an expected location and possibly an expected class. In order to provide bottom-up object detection for initialising the HLS, a detection mechanism is needed. Eq. 3.25

is expanded into

$$P(c, O|E, x) = P_{\mathcal{L}}(c|E, x) \times P(O|x), \tag{3.28}$$

where

$$O = \begin{cases} 1, & \text{there is an object at } x \text{ with } c \in \mathcal{L} \\ 0, & \text{otherwise} \end{cases} \tag{3.29}$$

For any potential object detection, $P_{\mathcal{L}}(c|E, x)$ represents the class probability given that the object is there, and $P(O|x)$ models the detection uncertainty. Given a detector which can provide a potential object center $x$ and a probability $P(O|x)$ that there is an object of a class $c \in \mathcal{L}$, this equation gives the probability that there is an object of a certain class at the given location.

One influential approach to model-based object detection is having a full joint probability distribution showing the spatial correlation of all the parts, as introduced by [Fergus et al., 2007]. This approach only works well with a small number of parts (3-7 are cited), and is computationally prohibitive for hundreds of categories consisting of dozens of compositions, as described here. A very popular method is using the Generalised Hough Transform, where each individual patch stores the offsets from the object centroids during learning. During detection, each patch votes for the potential object centres. Subsequently, the density is estimated in the voting space, and local maxima serve as object centroids. [Leibe et al., 2004] used this approach with a discrete set of class-specific patches for detecting cars and pedestrians. Since image patches are classified using an object-specific codebook, different centre locations can be grouped together for each codebook entry. This is different from the approach taken by Ommer and Buhmann, where the codebook entries are not class-specific, and the classification is based on compositions of patches, described by a vector of continuous values. Using a Hough voting scheme for continuous composition descriptions would require estimating $p(s_j|g_j)$, which is difficult.

Since the composition relevance function $P(\chi_c|g_j, s_j)$ is estimated during learning, a simpler approach is possible, based on randomly sampling points $x$ in the image. For each $x_n$, all $g_j \in \mathcal{R}_{x_n}$ within 50 pixels of $x_n$ are collected and the detection strength calculated as:

$$P(O|x_n) \propto \sum_c \sum_{g_j \in \mathcal{R}_{x_n}} P(\chi_c|g_j, s_j)$$

It is important to note that this approach is not meant to detect all objects in a given scene in a purely bottom-up manner. It aims at providing a set of very certain bottom-up detections which can initialise the high-level system and obtain updated priors or updated top-down hypotheses which are then used to disambiguate the remaining evidence as a part of a stepwise interpretation process.

#### 3.3.3.3 Top-down Grouping of Compositions

An important aspect of the hierarchical object recognition approach described in this section is that it improves classification accuracy by using *top-down grouping* of detected compositions in order to select the most relevant compositions and compositions of compositions. Equation 3.24 is then evaluated on the relevant compositions gathered in this step.

In the first step, random compositions (usually 120) are sampled from the area around a potential object centre. Then a preliminary classification is performed using the average descriptor of the collected compositions and all the compositions and their offsets:

$$P^*_{\mathcal{L}}(c|E) = exp\left[lnP(c|g^I) + \sum_{g_j \in \mathcal{R}} lnP(c|g_j, s_j)\right]$$

This will give a ranking of the classes in terms of classification confidence. A subset of the classes with good ranking (typically 10% of all classes) is selected as working hypotheses. At this point, it should be noted that the original approach of [Ommer and Buhmann, 2010] ranks on $P^*_{\mathcal{L}}(c|E)$ directly, assuming that classes are equally distributed. Taking the real class prior into account (such as a contextual prior provided by a high-level system) could be already used at this point, using the same approach as described in Section 3.6.2.2.

After a subset of classes is selected, only the compositions which have a high relevance for one of the remaining classes are kept (defined by a relevance function $P(\chi_c|g_j, s_j)$, typically 50% of the original set. Then compositions of compositions are randomly sampled and also filtered for relevance (using $P(\chi_c|g_k, g_l, r_{kl})$) in the same way. The result is a set of patch compositions and compositions of compositions which are relevant for the given classification problem, discarding irrelevant evidence, background and clutter.

This is an example of classification and evidence grouping working together in a two-step algorithm, improving the classification as a result. This is very important from the point of view of a middle-layer, because it provides an opportunity to not only influence the classification of collected evidence, but to also affect the way the evidence is grouped together, as will be explained in Section 3.6.

### 3.3.4 Modelling Detection Errors

There are two main sources of errors produced by image processing modules which affect the interpretation of the scene. The first one is the inaccurate detection of the position and object boundaries in the image. The second one is the detection of spurious objects, or false positives.

In order to measure detection errors, the output of each detector is compared to a set of hand-annotated images. Each detection which overlaps with an annotated object to a certain degree (for example, according to the PASCAL criterion) is considered to be a valid detection, and given the label of the annotated object. Each detection with no sufficient overlap with any annotated object is labelled as "False Positive". The learning of the probability distribution $P(C|E)$ is performed on this new set of data. This means that $P(C|E)$ is learned to reflect the phenomena typical for each detector separately. For example, some detectors deliver rectangular detections, others deliver convex polygons, so shape descriptors and the pixels used to calculate the feature vector will be different. This also allows learning the appearance of the false positive class.

#### 3.3.4.1 Modelling False Positives

Section 3.3.3.2 described the model for false positive detections when using patch-based models trained on a subset of all classes $\mathcal{L}$. Here, the concept is extended to region detectors and specialised image processing modules which deliver regions corresponding to objects. The prior probability that a detection at position $x$ is a false detection is $1-\sum_{c\in C} P(c|x)$, where $C$ is the set of all classes from the high-level ontology. Adopting the notation used for object detection using hierarchical object models,

$$P(c, O|E, x) = P(c|E, x) \times P(O|x),$$

where $P(O|x)$ is the probability that there is an object with class $c \in C$ at a given position in the image. In the case of hierarchical object models where the middle layer looks for potential objects, it was modelled as a function, based on the response of the object detector at that point. Experiments with several IPMs which deliver complete regions showed that false positive detections are not correlated with absolute image coordinates, so in that case $P(O)$ can be determined as a constant:

$$P(O) = \frac{\texttt{\# true detections}}{\texttt{\# all detections}}$$

The prior probabilities for classes from $C$ are dependent on the absolute position in the image, but this prior can be provided by a probabilistic high level.

The classification proceeds as before, using 3.18 for all classes in $C$ plus the false positive class "fp". For the false positive class, the initial prior $P(c = \texttt{fp})$ is equal to $1 - P(O)$ and the updated probability $P'(c = \texttt{fp}) = 1 - \sum_{c\in C} P'(c)$.

#### 3.3.4.2 Modelling Detection Accuracy

As mentioned before, image processing modules have different precision and may provide detections as boundaries accurate to the pixel level, or as rough blobs. The middle-layer

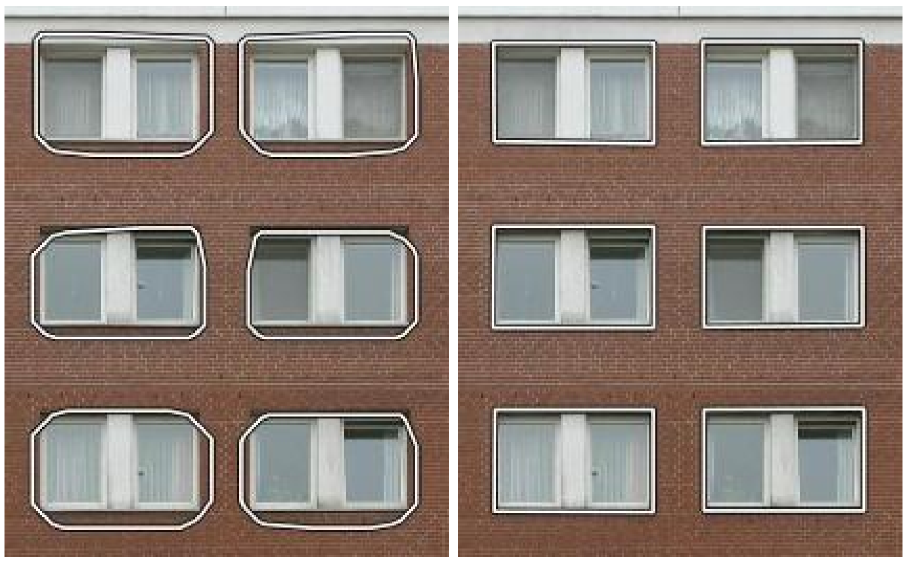


FIGURE 3.8: Size and position detection error of an IPM. Left: interest region detector of [Jahangiri and Petrou, 2008]. Right: manual annotations. Such detections can serve as evidence for semantically meaningful objects, but since the contours are inacurate, they might violate high-level size and position constraints.

architecture described in this chapter not only delivers classified object detections along with the detection confidence, but can also pass quantitative information about the detections, such as the position and size of the object in pixels, which can be used by the high-level system to reason about other objects in the scene.

Figure 3.8 shows the output of a region detector compared to the ground-truth annotation. While high-level constraints are often learnt from annotated data, which is reliable, the actual detections vary in terms of position and size of the bounding box due problems inherent in image processing. In practice, this leads to problems such as detections which violate high-level constraints relating to position and size of scene objects. A common example are man-made scenes, where scene objects (such as windows) are often nicely aligned, leading to elegant alignment constraints describing higher-level aggregates. An imperfect detection, whose location and bounding box do not correspond closely to the object in the image, can violate such constraints.

In order to provide reliable object detections corresponding to a high-level ontology, the middle layer not only needs to model the classification uncertainty (expressed as the posterior $P(C|E)$), but also the position and size uncertainty of each detected object. Let the centre of a detected object be represented by a vector-valued variable $x_o$. Then $x_o$ is related to the centre of the detected region $x_d$ through a probability distribution $P(x_o|x_d)$. Similarly, let $s_o$ represent the width and height of the object's bounding box. It is related to the (possibly error-prone) detected bounding box $s_d$ through $P(s_o|s_d)$. The two probability distributions are assumed to be independent in $x$ and $y$ directions and are approximated by four histograms by comparing the object detection (detected regions sufficiently overlapping with an annotated object according to an overlap criterion) to the annotations on a set of images.

In the case of some probabilistic interpretation systems, such as the Bayesian Compositional Hierarchies, the position and size of the objects in the image is already represented as a probabilistic distribution. Gaussian distributions are often used because they can be handled efficiently. In this case, a Gaussian curve is fitted to the histogram, and passed to the high level. Figure 3.9 shows the histograms of the $x$ and $y$ offsets of detection centres of the image processing module used for Figure 3.8. The x-axis shows the offset in 10-pixel steps, centered around zero. It was obtained by comparing the detections to manual annotations on a set of 160 images from the façade domain. It can be seen that the histograms can be well approximated by a normal distribution. Not all probabilistic high-level Systems model the position of the object using probabilistic distributions. For example, Conditional Random Fields of [Heesch and Petrou, 2009] only model class probability using neighbourhood relations. However, if a HLS models the distribution of the expected object positions explicitly (as is the case with Bayesian Compositional Hierarchies described in [Neumann, 2008]), it can directly make use of this information.

Crisp interpretation systems do not use probabilistic values. In order to express the detection accuracy, position and size (and therefore also the corners of the bounding box) are passed to such systems as intervals of possible values. Some high-level interpretation systems (such as SCENIC) can deal with such intervals directly. The range of the intervals depends on the detector used, as some detectors are more accurate than others. The four intervals are set to cover $\mu \pm 2\sigma$ of the Gaussian curves used for the probabilistic systems, which covers about 95% of the expected detector variation. The range information is passed together with every detected object and can be further restricted by high-level constraints of a crisp high-level system.

In both cases, the addition of evidence to the interpretation restricts the position and size of the object, but does not set it to a certain value due to the inherent uncertainty of the detector output. The more precise a detector is in terms of delivering an exact bounding box, the sharper the peak of the Gaussian curve passed to the probabilistic reasoner, and the smaller the interval passed to the crisp reasoner.

### 3.3.5 Combining Image Processing Modules

Since the middle layer allows for running several different image processing modules on the same image, one can use the outputs of different algorithms as evidence for the same interpretation. Since different algorithms have different strengths, and some specialised detectors are useful for a small set of classes but can not explain all the objects in the image, they need to be combined with more general (but less accurate) detectors.

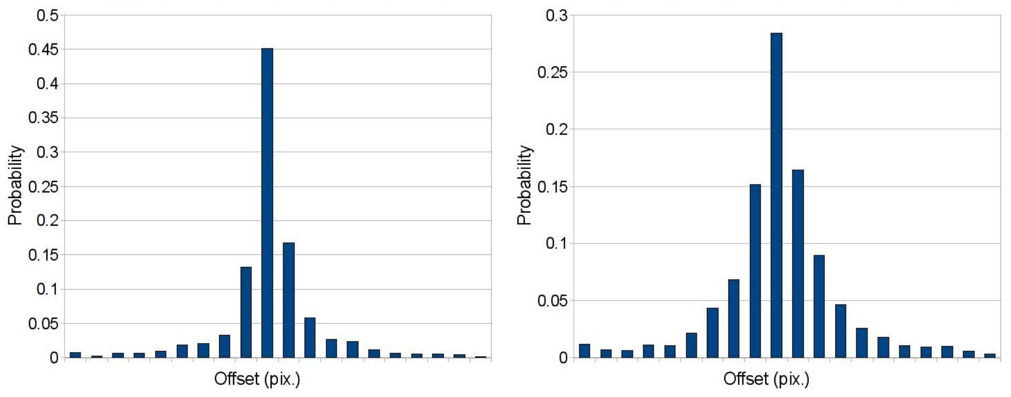


FIGURE 3.9: Detection centre uncertainty expressed as histograms of horizontal (left) and vertical (right) offsets, compared to manual annotations. The histogram bins correspond to 10-pixel offsets, centered around zero

Low-level detectors use and provide different features, and will give evidence which is not directly comparable. For example, see Figure 3.10 for a comparison of two different region segmentation schemes. The first one provides blobs which roughly correspond to objects, while the second one gives a very fine segmentation, down to the pixel level. The detected regions will influence the corresponding feature vectors, by affecting the shape of the region, the average colour of the region etc. Furthermore, different classes of detectors introduced in this section may have different representations of feature vectors, so a single probability distribution modelling all detections is not possible. Because of this, a separate estimate of $P(C|E_a)$ is needed for each detector.

It is tempting to try to combine the results of the individual detectors in order to boost performance, as is done in sensor fusion with different input channels. However, since the inputs (the image in question) are the same for all detectors, the results of IPMs are often highly correlated, so a naïve Bayes assumption does not hold and would not provide accurate probabilities. A joint probability distribution for a combination of feature vectors from two detectors, however, could lead to improvement in some cases, provided that there are enough samples to learn the more complex $P(C|E_1, E_2)$, $E_1$ and $E_2$ being the feature vectors describing two overlapping detections. The feasibility of this approach was investigated using three detectors on a set of images from the façade domain.

An experiment comparing the results of three separate detectors[3] with an annotated database of façade images showed no statistically significant improvement in object detection reliability when a region was detected by several detectors simultaneously. Although detections of windows and other façade elements were often correlated, so were many of the misdetections (false positives). This is not as surprising as it might seem at first: two methods used are region segmenters which react to strong edges, and the third is a specialised detector built from Haar-like features, which also appear along

[3]an AdaBoost-based window detector based on [Šochman and Matas, 2009], the stable region detector of [Drauschke, 2009] and the interest region detector of [Jahangiri and Petrou, 2008]

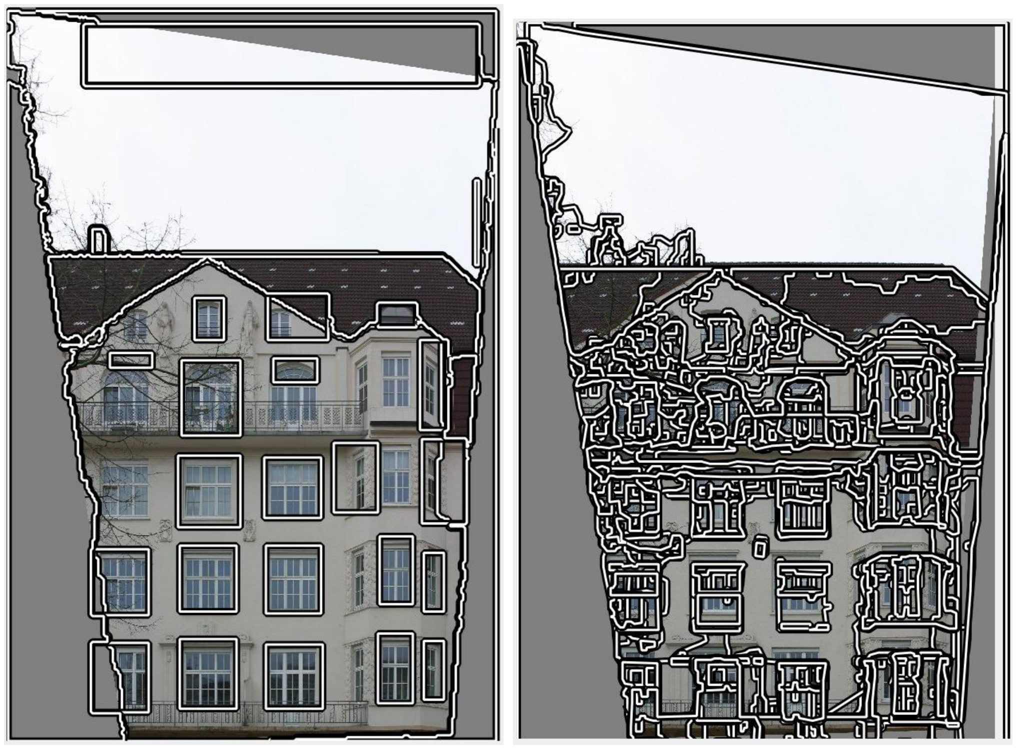


FIGURE 3.10: A comparison of two region-detection algorithms. Left: Interest region detector of [Jahangiri and Petrou, 2008]. Right: Stable Region Detector of [Drauschke, 2009].

edges and corners. As a result of this, it is difficult to formulate a generic method for fusing evidence from different detectors in a way which would improve the classification or detection reliability.

It is, however, still possible to exploit the fact that some detectors are better at detecting objects of certain classes. Let $S_R$ be a set of overlapping regions and $S_E = \{E_1, E_2, \ldots, E_J\}$ a set of feature vectors describing the regions in $S_R$, $|S_R| = |S_E| = J$. The classification problem is formulated as:

$$C = \arg\max_C P(C|E_j), E_j \in S_E. \tag{3.30}$$

As soon as a high-level view is instantiated and attached to the region with the highest probability, all the other regions in $S_R$ are ignored. The result is an interpretation which picks the most certain evidence from an evidence soup, discarding the remaining regions as clutter.

As long as the probability function $P(C|E)$ accurately models the underlying probability, the classifications provided by them will be comparable. This is not necessarily the case for different types of detectors (see the beginning of this section), as they model evidence differently. The structured models built from image patches in particular introduce a number of simplifying assumptions.

If $P(C|E)$ is estimated on the same data for all IPMs, the class probabilities are comparable across the detectors.Section 5.4.3 describes an experiment combining three detectors where the probabilities were estimated on the same training set.

## 3.4 High-Level Integration

As described in Section 3.2, the interface between the low-level and high-level is modelled using intermediate objects called views. The mapping from signals to symbols therefore has three levels: Evidence, Views, and Scene Objects. The low-level integration described in the previous section allows for grouping and classification of evidence to the level of views. Such views represent a natural interface to high-level system and can be used to pass information in both directions: bottom-up, in the case of classifying and integrating new evidence, and top-down, in the case of passing high-level expectations about objects which have not yet been observed.

The addition of views also makes it possible to construct complex high-level models which represent scenes in the more natural 3D environment, and not in terms of pixel distances. This makes the interpretation extensible to more general cases, where the location of the camera is not restricted, and scenes can be observed from different angles and distances, and under a range of illumination conditions.

### 3.4.1 Views as an Interface to the High-Level System

The most basic example of using views as an interface is bottom-up detection of objects, which is typically how an interpretation process is started. The middle layer receives evidence from image processing modules and detects objects defined by the domain ontology and passes views with high detection confidence to the high-level. Such a view signifies that a certain object from the scene model was observed at a certain position in the image. Alternatively, the high level can create expectations and pass them down to the middle layer in terms of view hypotheses. Such a view signifies that a certain object is expected at a certain location. In this way, views serve as a generic means of data exchange between the high-level and the middle-layer. The middle layer and the IPMs do not know how the hypotheses were generated, and the high level does not need to know how the objects were observed.

Since the interpretation is seen as a stepwise search, the state of both the middle layer and the high-level must be persistent, and both must have access to the current state of the interpretation. This is because the model is adapted to the data during the interpretation process, so the views created at a previous step may need to be adapted in the future. There are two examples where this can happen:

- the middle layer creates a view, but the position and size have a degree of uncertainty due to the detection inaccuracy of the IPM (see Section 3.3.4. The view is integrated into the interpretation, and high-level constraints reduce this uncertainty.
- the HLS creates a hypothesis with allowed value ranges for position and size. Such a hypothesis is matched to low-level evidence by the middle layer, and the value ranges are set to match the evidence that the view was matched to.

In the proposed architecture, this means that both the middle layer and the HLS must have access to all the views observed during the current interpretation. Section 4.2.1 discusses a practical implementation of such an interface.

The classification framework introduced in Section 3.2 made the assumption that the appearance of a view in the image $E_a$ is independent from the structural information about the object $E_s$ given the knowledge of the class. The consequence of this assumption was that it is possible to use appearance for classification, and the structural information for providing dynamic priors from the high-level. Therefore, the middle layer proposed in this thesis deals with the class probability estimates as described in Section 3.3, and the high-level reasons based on the class and structural information (such as position and size). This defines the information about views which must be shared between the middle layer and the high level:

- **position**, e.g. the view centroid or the middle of the bounding box,
- **bounding box** describing the region the object view covers in the image, and
- **class** of the view.

Passing more detailed information is possible, for example passing bounding polygons instead of bounding boxes, but since bounding boxes are very common in both high-level systems and image processing modules, they were used throughout this work.

This information allows the high-level system to propagate constraints and pass down expectations based on the positions and classes of the observed objects in relation to each other, while allowing the middle layer to model class probabilities for IPMs.

The following sections specify the integration requirements for high-level systems. Since crisp systems and probabilistic systems operate differently, they are described separately. In each case, the bottom-up and top-down information flow is described.

### 3.4.2 Integration with Crisp High-Level Systems

Crisp high-level systems do not deal with probabilities, so another way of describing the classification uncertainty and detection error is needed. The solution described in this

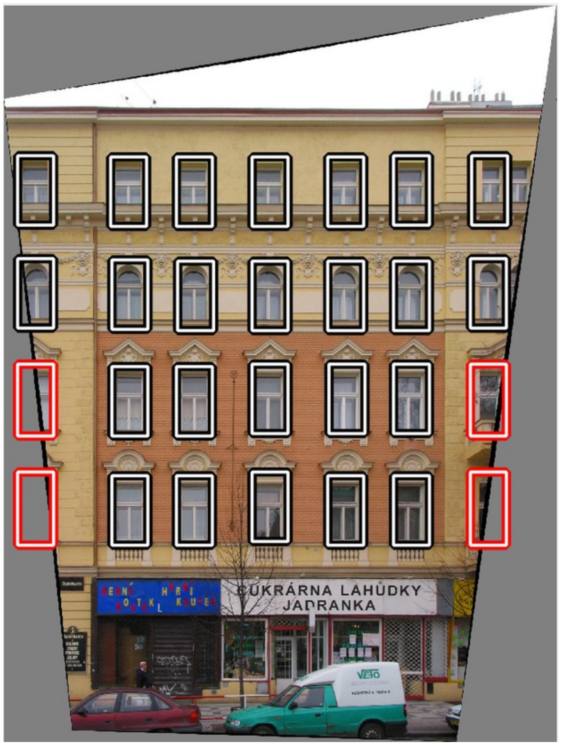


FIGURE 3.11: High-level hypotheses generated by the SCENIC system on an image from the façade domain. The black and white rectangles show the evidence detected by the structured window detector of [Terzić et al., 2010], which were passed as views to the high level during previous steps. The red rectangles show where the high level expects further windows, based on high-level constraints.

thesis, which was implemented as a part of the SCENIC scene interpretation system, is to represent this uncertainty as a set of allowed value ranges. Each view passed to the high level contains the following information:

- **position** expressed as x and y value ranges: $p = \langle x, y \rangle$; $x, y = \langle min, max \rangle$; $min, max \in \mathbb{R}$,
- **horizontal and vertical size** of the detected view expressed as value ranges: $s = \langle w, h \rangle$; $w, h = \langle min, max \rangle$; $min, max \in \mathbb{R}$,
- **bounding box coordinates** expressed as value ranges: $B = \langle tl, br \rangle$; $tl, br = \langle x, y \rangle$, with $x$ and $y$ being ranges as before,
- **possible classes**, expressed as a set of tuples $\langle class, probability \rangle$

In many cases, only the strongest class is needed, but it might be useful to pass several classes having high probabilities to the high level, in the case certain classes are visually similar and an exact classification is not possible. For example, a tanker and a loader in an airport refuelling scene have similar appearance, but the scene context, determined by other observed objects, might make only one of the two possible.

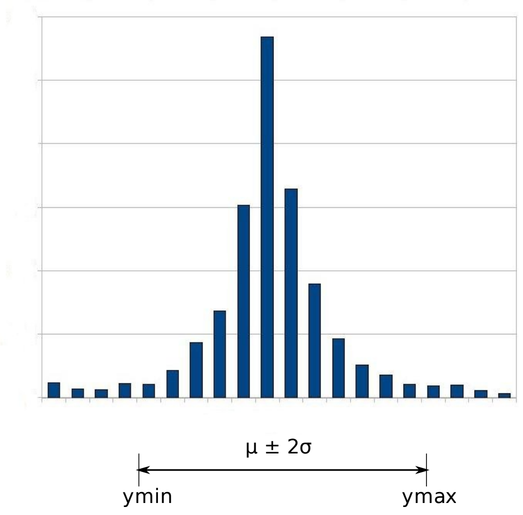


FIGURE 3.12: Expressing detection uncertainty as a value range. Shown on the top is the distribution modelling the positional uncertainty of an IPM along the y axis. Below it is an interval covering two standard deviations around the mean. The Gaussian approximation of the distribution is passed to the probabilistic high-level systems, and the interval is passed to crisp high-level systems.

The passing of top-down expectations down to the middle layer is the same in this case. A view hypothesis is passed down. The value ranges for the parameters are restricted by high-level constraints, which restricts the evidence to which the hypothesis can be matched. Therefore, in the case of a crisp interpretation system, the data exchange between the high level and the middle layer is always performed as an exchange of view objects. The integration required is that the high-level system can (i) accept views in this format during the interpretation, (ii) pass hypotheses to the middle layer in this format during the interpretation, and (iii) ask the middle layer to confirm a hypothesis that was passed down.

### 3.4.3 Integration with Probabilistic High-Level Systems

In the case of probabilistic high-level systems, the bottom-up step is the same as with the crisp high-level systems: a view is created by the middle layer, and passed to the high level. The important difference is that, instead of value ranges specifying the position and size uncertainty, the parameters of the normal distribution calculated in Section 3.3.4 are passed, giving:

- **position** expressed as Gaussians over x and y coordinates: $p = \langle x, y\rangle$; $x, y = \langle \mu, \sigma\rangle$; $\mu, \sigma \in \mathbb{R}$,
- **horizontal and vertical size** of the detected view expressed as Gaussians: $s = \langle w, h\rangle$; $w, h = \langle \mu, \sigma\rangle$; $\mu, \sigma \in \mathbb{R}$;

- **bounding box coordinates** expressed as Gaussians: $B = \langle tl, br \rangle$; $tl, rl = \langle x, y \rangle$, with $x$ and $y$ being distributions as before,
- **possible classes**, expressed as a set of tuples $\langle class, probability \rangle$

Probabilistic high-level systems do not necessarily create hypotheses. Because of this, an alternative way to pass high-level expectations down to the middle layer is needed. The middle layer requires an updated contextual prior for all classes at a certain location of the image in order to improve the classification of evidence. The high level needs to implement a query mechanism by which the middle layer can obtain this information. Assuming that the high level approximates the JPD described in Section 3.2, it needs to accept a bounding box describing a section of the image, and return the class probabilities for all classes in that area, considering all previously observed views.

It should be noted that not all probabilistic high-level systems perform a stepwise integration of evidence in order to approximate the JPD. For example, the CRF approach of [Heesch and Petrou, 2009] takes all regions in an image as input, and arrives at class probabilities for each region by stepwise probabilistic relaxation. The interface to the middle layer was realised in the same way using this system, with the middle layer passing a bounding box for each detected region and asking for class priors. In this case, the best evidence is first determined in a purely bottom-up manner, and the high-level acts to disambiguate the classification by supplying contextual class priors.

## 3.5 Integrating Evidence

The essential tasks required for integrating a particular group of evidence, such as modelling image processing modules, grouping and classifying evidence, and using high-level knowledge, have all been described in detail in this section. Since stepwise interpretation is used in order to reduce the combinatorial complexity, the order in which evidence is integrated is important.

The simplest approach is to group all evidence in a bottom-up fashion, and pass it to the high-level system in one step, letting the high-level system deal with the complex assignment problem. Heesch and Petrou [2009] use such an approach. Their Conditional Random Field receives the structural information (positions and polygons describing the views) from the middle layer and assigns labels to the regions based on the pairwise spatial relations between regions. Probabilistic relaxation is used to arrive at the final set of probabilities for each label and each region. Such probabilistic information is based on structural information only, and can therefore be used as a contextual prior by the middle layer and combined with the appearance-based evidence, as described in Section 3.2.

The problem with this approach is that it assumes that the detections are accurate. Since the scene model (expressed in terms of pairwise spatial relations between regions, with no hierarchy) is learned on annotated data with no errors, its performance decreases when used on imperfect detections. The problems include false positives and false negatives, as well as detections which are either too big or too small to the point where the spatial relations with other objects are affected. A stepwise approach which integrates detected views one by one can help combat the low-level problems by supplying scene context and helping to detect false detections.

### 3.5.1 Integration Sequence in a Stepwise Interpretation

Stepwise interpretation attempts to find the best interpretation through a series of evidence interpretation steps. After each step, evidence is propagated in the high-level system (either as crisp constraints or as probabilistic correlations), and used to affect further integration steps. This is a greedy approach which reduces the complexity, but does not guarantee arriving at a global maximum. Early assignments are particularly problematic, as the high-level context has not been established yet. Experiments in Section 5.4.2.1 show that poor assignments early in the interpretation process tend to result in wrong context being established, and lead to more wrong assignments later.

There are several ways to deal with the issue of bad assignments caused by too much uncertainty from the low level. The first one is parallelism, where alternative assignments are pursued in different threads at the same time. Poor assignment decisions will lead to constraint violations after a while, and such threads die, leaving only good interpretations alive. In [Neumann and Terzić, 2010], beam search is suggested to reduce the effects of the resulting combinatorial explosion. An alternative to this approach is treating interpretation as a search problem, where alternative decisions are investigated in turn, and backtracking is performed if a conflict arises. [Hotz and Neumann, 2005] exmploy such a model.

### 3.5.2 Confirming High-Level Hypotheses

The major task of a middle layer in an interpretation system is to compare the model-based expectations from the high-level system with the observed low-level evidence. The state of discrepancy between the expected and the observed is also called a **conflict**. More specifically, a conflict between the evidence and interpretation occurs when an object hypothesis exists in the high-level system which overlaps with evidence to which it cannot be matched, after the detection uncertainty has been taken into account. There are two possible explanations for such an event:

- the interpretation is wrong and not supported by the image, or

- the interpretation is correct, but some of the expected objects are **occluded** by other objects, or have not been detected correctly (false negatives).

A common source of incorrect detections are low-level phenomena, such as partially visible objects at image boundaries (leading to wrong size and classifications) and incorrect segmentations, resulting in wrong object boundaries of merged object detections (several objects detected together). Such errors are modelled for each image processing module separately, and the accuracy evaluated on a training set, as described in Section 3.3. The middle layer takes the position and size accuracy into consideration when checking if detections can overlap.

The problem also appears when combining evidence from different detectors, or when using a detector which delivers evidence detected at different scales. The middle layer will typically have sets of overlapping regions, and only one can be matched to a high-level view. Instantiating several objects of the same class at the same location would violate the high-level model.

Since the role of the middle layer is to mediate between the high-level system and the image processing modules, it needs to provide a mechanism for differentiating between legitimate occlusions and conflicts. This section describes a generic, ontology-based approach to detecting occlusions implemented within the SCENIC interpretation system. The middle layer has access to the knowledge base used to interpret the scene, and the knowledge base defines which objects can occlude each other in a typical scene. Even after accounting for low-level phenomena such as IPM errors and image boundaries, there are still several reasons why two *correct* detections might overlap in a typical image without causing a conflict:

- one object partially occludes another, such as crowded pedestrian scenes, or vegetation covering a part of a building, or
- one object is a part of another, such as a detection of a window and a detection of an entire building, obtained at different scales.

The knowldege base models the first one in terms of a “can-overlap” relation, and the second one in terms of a partonomical hierarchy. If the middle layer has access to the knowledge base, it can use this information to distinguish between allowed occlusions and actual conflicts. The approach with a high-level relation modelling possible overlaps is similar to the conflict management in the early SIGMA system [Nagao et al., 1979], which also uses partonomy and high-level conflict relations to reason about conflicting assignments.

Therefore, at each integration step, the middle layer has to examine whether the most likely classification of a potential new view conflicts with an already confirmed object. In order to do this, it evaluates $P(C|E)$ using the context-based classification equation, and obtains the most probable class for the view. It then finds all the already instantiated views which overlap with the new view and tests whether they are allowed to overlap by checking (i) for potential partonomical relations between the two classes, and (ii) for a "can-overlap" relation between the two classes. If the overlap is allowed, then one of the views is taken to occlude the other in the image. If the overlap is not allowed (e.g. a door and a window can not overlap and are not parts of each other), the two views are in conflict. In order to resolve the conflict, the interpretation system can ignore the new view as spurious evidence, leave the classification for a later stage when class priors have changed, or remove the older view and integrate the new one. The middle layer can deal with all three possibilities and does not prescribe what should be done in such a case.

A special case of the conflict problem occurs if the high level creates a view hypothesis and asks for it to be confirmed, as is the case with our interpretation system SCENIC. In this case, not only must the middle layer try to match the hypothesis to evidence, but also report to the high level whether it was successful.

Take a crisp interpretation system interpreting an image from the façade domain as an example. The evolving interpretation creates a hypothesis of a window view in the image. The first possibility is that the hypothesis is wrong, and something different was observed, for example there is evidence suggesting sky at that position. Since windows can not float in the sky, the window hypothesis is wrong. The other possibility is that a window was predicted, but a tree was observed instead. It is possible that a tree is in front of the window (this is modelled by a "can-overlap" relation between a tree and a window), which makes it an occlusion. There is no evidence supporting the hypothesised window, but the interpretation may still be correct. The third possibility is the easiest of the three, in the case that a window was expected, and a window was also observed.

Based on this, three states can be defined when attempting to match evidence to a high-level hypothesis:

- **Confirmed**: the evidence fits into the high-level interpretation,
- **Refuted**: the evidence is in conflict the interpretation, and
- **Don't know**: there is no evidence for or against.

"Confirmed" state is the result of successfully integrated evidence. "Refuted" means that the evidence disproves the interpretation, and represents a conflict which must be dealt with in the high level. "Don't know" means that nothing was matched, either because the object was not detected by any image processing module, or because an unexpected object was detected instead, but which does not conflict with the interpretation.

As before, the actual interpretation of what was observed in the image is probabilistic. The hypothesis is passed down as a view covering an area in the image and carrying the expected class. The area covered by the view defines the centre of the view in the image and contains some evidence $E$, which allows the middle layer to calculate class probabilities $P(C|E)$ based on the evidence. If the most probable class based on the evidence matches the hypothesis, the hypothesis is confirmed. If the most probable class conflicts with the hypothesis as described in the previous section, the hypothesis is refuted. If the classes do not match, but are allowed to overlap, then a "don't know" answer is given. If there are overlapping detections at the low-level, the one with the highest classification confidence is used, as described in Section 3.3.5.

### 3.5.3 Integration of Evidence with Probabilistic Systems

In this section, an algorithm is presented which integrates the evidence into a probabilistic interpretation using a sequence which minimises the error introduced at each step. At each step, the class prior from the previous step $P(C)$, is replaced by an updated class prior $P'(C)$, which reflects the scene context, and Equation 3.18 is used to calculate the new posterior class probability:

1. initialise $P'(C)$ to the domain priors $P(C)$
2. select the evidence (region or a group of patches) with the highest classification confidence $P'(C|E)$
3. make the most probable evidence assignment and propagate the evidence, obtaining updated $P'(C)$
4. repeat Steps 2-3 until all evidence has been assigned, or all variables of the model explained by evidence

The algorithm will stop once no further evidence assignment is possible. This means that either all variables of the model have been matched to available evidence, or that no further views can be instantiated from the remaining evidence, which would fit into the current interpretation.

### 3.5.4 Integration of Evidence into a Crisp System

Since crisp interpretation systems do not update the class priors, the scene context must be propagated using top-down predictions, or hypotheses. The evidence is integrated using the following algorithm:

1. integrate the most likely piece of evidence using domain priors $P(C)$

2. repeat Step 1 until hypotheses are obtained from the high level
3. match the hypotheses to evidence
4. if there is unmatched evidence remaining, return to Step 2

The algorithm terminates when all the available evidence has either been matched, or explained as false positives or occlusion.

### 3.5.5 Structured Evidence

In scene interpretation, structure can be manifested both in terms of compositional structure, modelled at a high level using a compositional hierarchy, and in visual structure, where repetition is detected in the image at a visual level. The structure inherent in the image can be used by an image processing module to improve its performance, especially in highly structured domains. [Terzić et al., 2010] shows a low level detector which uses the assumption about horizontal and vertical alignment of windows in a scene to grow multiple 2D structures from individual seeds, and where the high-level is used to merge these structures into a coherent interpretation. The details of that work are specific to the particular application, so the experiment is described in more detail in Section 5.2.2. This section will only introduce the changes needed to the presented architecture which are needed to enable the passing of low-level structure to the high level.

Figure 3.13 illustrates the process using a real example. The low level detects both the regions corresponding to the windows (which is represented as window views in the middle layer) and the aggregate consisting of the individual views. The high-level knowledge base models a “Window-Array” concept which can contain façade elements and integrated into a façade. The middle-layer receives the part-of relation between the individual windows and the formation from the low level. When passing the views to the high level, they are passed at the same time, and the partonomy between them is expressed through the new “has-view-elements” and “view-element-of” relations. The high-level system receives this information, which simplifies the role assignment problem, and integrates the views into the interpretation. The additional relations must be added to the view description passed to the high level. They can be modelled as arrays of IDs.

Section 5.2.2 shows how the combination of low-level and high-level structure sources performed in this way can improve the overall interpretation.

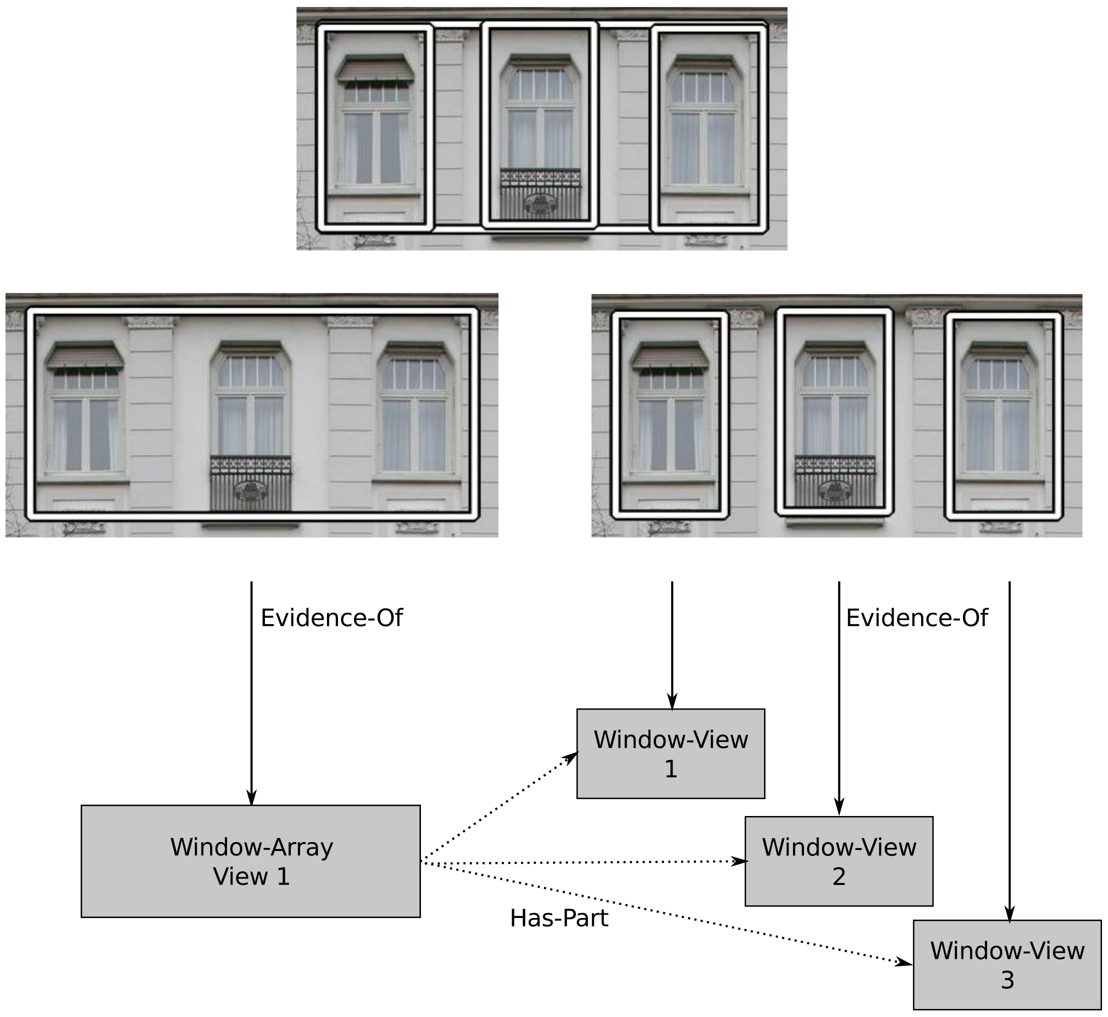


FIGURE 3.13: Passing structured evidence to the high level. The top image shows the overlapping detections supplied by the low level. They consist of three window detections and one bounding box for all detections which fulfil a set of spatial constraints (in this case, horizontal alignment). The two images in the middle show the two different kinds of detections separately. The structural detection is matched to an existing high-level concept "Window-Array", and the individual parts are matched to the instances of "Window". The diagram at the bottom shows the information passed to the high-level. Notice that the part-of relationships are also included, as they were provided by the low level.

## 3.6 Feedback

So far, this chapter has presented the basic mechanisms needed for matching results of image processing modules to variables in a high-level scene model, and has argued for the need for top-down feedback. General concepts for implementing top-down feedback were introduced, such as dynamic priors and crisp hypotheses.

This section describes specific top-down feedback mechanisms in more detail. Since the information provided by the crisp and probabilistic high-level systems is inherently different, the two are described separately.

### 3.6.1 Feedback for Crisp Interpretation

Crisp interpretation systems can not provide prior probabilities for classes in order to assist classification. On the other hand, reasoning with crisp rules can limit the allowed ranges of parameters for scene objects. For example, if the location of a building in the image is known, this will limit the allowed locations for windows. The remaining uncertainty about position, size and other parameters can be expressed as a set of allowed ranges for parameters describing scene objects. As described in this chapter, this knowledge about yet unseen objects is passed down to the middle-layer in terms of "hypotheses" described by value ranges.

#### 3.6.1.1 Middle-Layer Evidence Grouping and Classification

If a hypothesis is passed down to the middle layer indicating where an object is expected, this can affect the middle-layer classification in several ways. First of all, the hypothesis specifies the class of the expected object, or more generally, a list of possible classes. The contextual MAP classifier introduced earlier in this section delivers a list of probabilities for all the classes it was trained for (if it is used with a crisp system, it uses static domain priors instead of contextual priors). Not all of the classes are allowed by the high-level constraints, and creating such a view would violate high-level constraints. The hypotheses offer a way to let the middle layer know this, so it can deal with such situations as described in Section 3.5.2.

If a hierarchical model is used, then evidence needs to be grouped before it is classified, as described in Section 3.3.3.2. View hypotheses offer a simple way to restrict the evidence one needs to consider by providing the expected bounding box of an object in the image. Instead of trying to detect meaningful combinations of patches in a bottom-up fashion, view hypotheses offer a natural grouping of patches and compositions. The hierarchical model can then give the class probabilities for the region provided.

In addition to the initial grouping of evidence, hypotheses can also serve to select relevant compositions as described in Section 3.3.3.3. Instead of making bottom-up working hypotheses by selecting a subset ofclasses based on the initial classification (Ommer and Buhmann used 10%), one can choose the set of classes allowed by the current high-level hypothesis. Then, compositions are specifically selected to try to confirm the hypothesis. The result is a grouping of compositions based on high-level expectations.

#### 3.6.1.2 Sensitivity Adjustment and ROI

Low-level algorithms are typically controlled by a number of parameters. The quality of detection often depends on selecting the best value of these parameters. Modelling all possible algorithms and their parameters generically is not possible, but if the algorithm provides one adjustable parameter which controls the ratio of false positives to false negatives, then this parameter can be used in a top-down fashion to try to detect unclear objects given a strong expectation from the high level.

Another important parameter is region of interest (ROI), which selects a subset of the image for processing. This can be useful with algorithms which with long runtimes. The high-level view hypotheses provide regions in which an object is being looked for and therefore act as an attention mechanism, allowing more efficient use of computing resources.

[Hotz et al., 2007] uses the concept of views to implement such a feedback loop. Given a number of window views from a specialised detector, the high level can predict the locations of the missing windows, which were not detected in the first step. In the second step, the detector is started again using the ROI specified by the view hypotheses, and an increased sensitivity, and manages to find the missing windows.

While this approach can be useful on simpler problems and using one detector, a proper integration of sensitivity tuning when using several detectors would require careful modelling of how the sensitivity of each detector affects the detection and classification rate, and was not further pursued in this work.

#### 3.6.1.3 Passing Down Expectations to Image Processing Modules

Some low-level algorithms use perceptual grouping of evidence to provide a purely bottom-up segmentation of the image. In such a case, low level evidence (edgels, interest points, superpixels, etc.) is not available at the middle level, and a direct influence on evidence grouping such as described in Section 3.6.1.1 is not possible. On the other hand, purely bottom-up grouping will not necessarily produce accurate segmentation without high-level feedback, so some form of downward information flow is necessary. A simple approach is to pass the size and ROI expectations, which will limit the size of the detected regions. However, more detailed control is desirable.

If decision trees are used for estimating class probabilities (as described in Section 3.3.2.1), a generic way of passing expectations is possible. Figure 3.14 shows a feature space partitioned by a decision tree learnt using the algorithm described in Appendix A. Each one of the leaves is associated with a probability vector $P(C|L)$. Since $P(C)$ and $P(L)$ are determined during learning, the probability distribution $P(L|C)$ can be determined. This distribution models the most likely appearance of a view belonging to a certain class. The leaves with high $P(l|c)$ are more likely to be observed for a given class $c$.

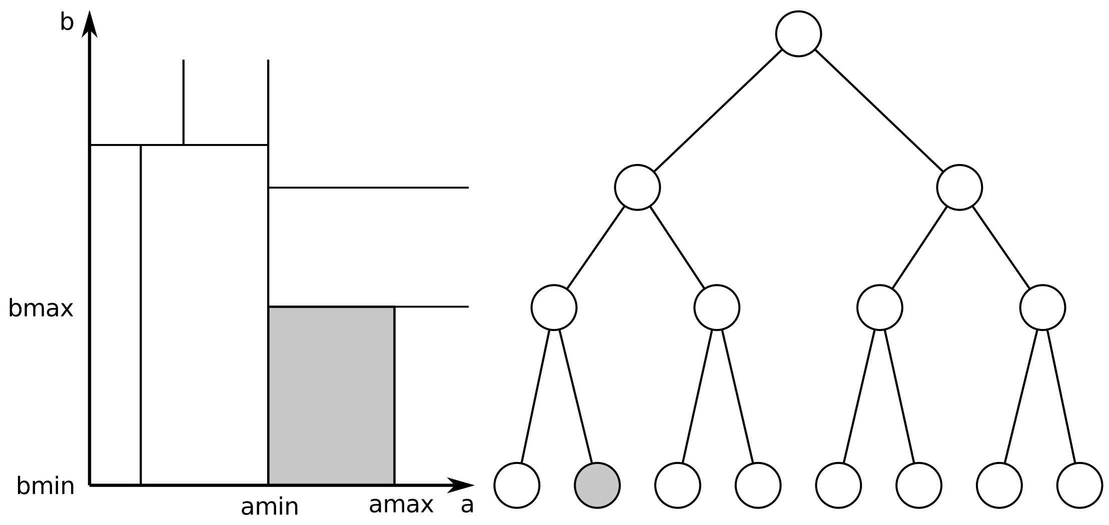


FIGURE 3.14: A 2D feature space divided into partitions by a decision tree. Each rectangle on the left side corresponds to a leaf of the decision tree shown on the right. Since the decision tree is univariate (decisions are based on one variable at a time), the partitioning is axis-parallel and can be expressed as a combination of horizontal and vertical value ranges.

An attractive property of univariate decision trees is that each leaf can be described as a hyper-rectangle in a multi-dimensional feature space. Figure 3.14 shows the example in two dimensions, which can be expanded to any number of dimensions. Therefore, each leaf can be described as a combination of one-dimensional ranges, determined for each feature dimension independently. Such descriptions can be efficiently stored and passed as parameters to an image processing module .

In order to do this, the IPM has to provide regions described by feature vectors, and calculate these feature vectors internally. Then, given a high-level hypotheses specifying the expected class, width, height and region of interest (bounding box of allowed positions), the decision tree modelling the class probabilities for this segmenter delivers the set of leaves which are typical for the expected class. This can be all leaves with $P(l|c) > 0$, but experiments show that classes in typical domains often overlap to such an extent that this might cover most of the feature space, especially if probability smoothing is used (as discussed in Section 5.3.3). Therefore, it is better to set a threshold above zero.

Each of the collected leaves is described as a one-dimensional range in each feature dimension, and the complete information is passed to the IPM. This is essentially a description of what to look for, described in terms the IPM can understand (as it knows how to calculate the used features). Instead of using general principles for perceptual grouping of evidence, the IPM tries to group the evidence in a way which conforms to one of the allowed appearances passed down by the middle layer.

This type of feedback is generic, because it does not presuppose anything about the IPM, other than being able to read the allowed feature ranges. However, it obviously requires the IPM detector to be modified to take advantage of the provided information.

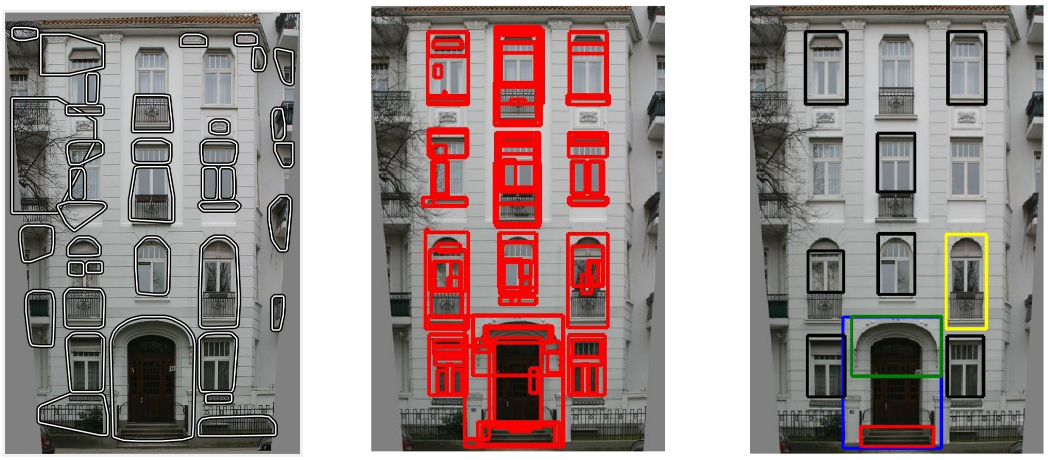


FIGURE 3.15: A proof of concept result showing feedback to an IPM illustrated on the detector of [Jahangiri and Petrou, 2008]. The left image shows the detected regions after a fully bottom-up grouping of smaller regions, based on heuristics. The middle image shows the small regions before merging. The right image shows the result of the region merging after high-level expectations are passed down in the form of allowed size, position and feature ranges. It can be seen that the top-down step improves the segmentation significantly.

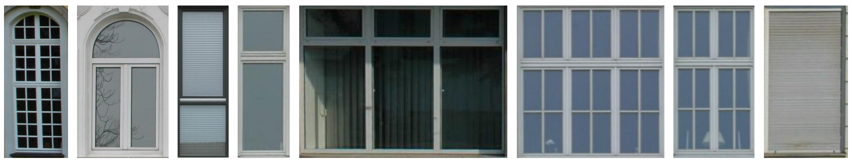


FIGURE 3.16: Some windows from the annotated database.

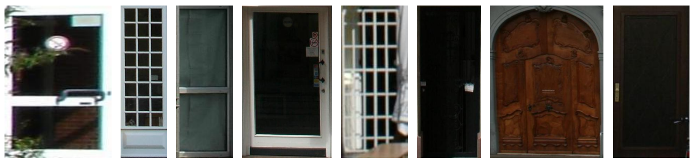


FIGURE 3.17: Some doors from the annotated database. The appearance and shape varies a lot and there is significant overlap with the Window class.

This was successfully performed with the detector of [Jahangiri and Petrou, 2008], but might not be feasible for all low-level algorithms.

### 3.6.2 Feedback for Probabilistic Interpretation

Probabilistic interpretation systems do not provide crisp ranges and sets of allowed values like crisp systems. They provide class posteriors given previously observed context from other objects. From the point of view of view classification, these are contextual priors which can be used to support the detection and classification decisions.

#### 3.6.2.1 Disambiguation Using Contextual Priors

The classification formula shown in Equation 3.18 allows for naturally integrating contextual priors at a middle level. It might not be obvious why updated priors are useful for the classification of object views, if all the relevant evidence was observed. However, in many realistic domains, the classification is difficult without context. Figure 3.16 shows some windows from the façade domain, and Figure 3.17 shows some doors. When these images are taken out of context, it is not always easy to distinguish between the two classes based on appearance alone. On the other hand, doors and windows have very different roles in a typical building, and a wrong classification (e.g. mistaking a window for a door) will lead to poor interpretations.

When using image processing modules which provide regions representing objects, this problem is further complicated by the fact that detection errors described in Section 3.3.4 introduce additional uncertainty, because the regions do not perfectly correspond to the object contours in the image, unlike the example given here. This means that object-related features are mixed with parts of the background or even parts of other objects, making the feature vector less discriminative. Experiments in Section 3.2.2 show that priors introduced this way can significantly improve the classification rate.

#### 3.6.2.2 Top-down Evidence Grouping

If a hierarchical visual model described in Section 3.3.3 is used to detect and classify objects, the contextual prior can also be used to affect top-down selection and grouping of compositions. In the case of crisp feedback provided by view hypotheses, this was done by restricting the search to compositions and groups which are relevant for the classes predicted by the hypothesis. If a prior is available, then this selection can be further influenced. As described in Section 3.3.3.3, the initial bottom-up hypotheses are provided by picking a subset of classes with a high posterior probability

$$P^*_{\mathcal{L}}(c|E) = exp\left[lnP(c|g^I) + \sum_{g_j \in \mathcal{R}} lnP(c|g_j, s_j)\right].$$

This probability assumes that all classes are equally probable. Since high-level contextual priors are generally available when using a probabilistic high-level system, this probability can also be affected by the contextual prior:

$$P^{*'}_{\mathcal{L}}(c|E) \propto exp\left[ln\frac{P(c|g^I)P'(C)}{P(C)} + \sum_{g_j \in \mathcal{R}} ln\frac{P(c|g_j, s_j)P'(C)}{P(C)}\right].$$

where $P'(C)$ is the updated contextual prior and $P(C)$ the prior representing the training set, typically $1/|C|$.

A subset of working class hypotheses is then selected using these updated probabilities, and the evidence selection and grouping is performed to maximise the relevance for these classes. This means that the high-level context is involved in the object detection at an earlier stage, affecting the evidence selection and grouping.

## 3.7 Bookkeeping

In scene interpretation systems involving time, keeping track of which object is associated with which evidence is very important. In the robotics community, this task is often called "anchoring" [Coradeschi and Saffiotti, 2000], and is important for turning abstract goals into motor movements of the robot. In scene interpretation, maintaining correspondences between symbols and signals is important to keep track of what is going on in the scene, and how new observations change available knowledge.

When interpreting static scenes, the assignments of low-level evidence to high-level views is also static. In other words, there is usually a single best set of assignments for each possible alternative interpretation of the image. However, due to the stepwise interpretation process, maintaining correspondences across individual interpretation steps is still necessary. There are several reasons:

- The interpretation system as a whole must remember which evidence and views were matched to each other in previous steps in order to provide a final complete interpretation.
- It is important not to match a view to a piece of evidence which was already assigned to another view during a previous step.
- It is important not to instantiate multiple conflicting views at the same position over several unrelated interpretation steps.

The bookkeeping keeps information about the current state of the mapping between the high-level concepts and the observed evidence in the scene, by representing matches as triples containing a piece of evidence (which can be composed of smaller pieces of evidence), the view matched to the evidence, and the matching confidence which is a probability between 0 and 1. It serves the role of memory, remembering the assignments made during previous steps.

### 3.7.1 Relation to Anchoring in Robotics

Keeping track of evidence-view correspondences is similar to perceptual anchors used in robotics, such as the work of [Coradeschi and Saffiotti, 2000]. As described in Section 2.2.8, the authors propose a formal definition of the anchoring problem. They define anchors in terms of maintaining correspondences between a symbol system $\Sigma$ and a perceptual system $\Theta$ through a set of perceptual signatures $\Gamma$. In this section, their formalism is used to describe the bookkeeping of correspondences throughout an ongoing image interpretation process.

The perceptual system $\Xi$ is represented by a set of percepts $\Pi = \{\pi_1, \pi_2, \ldots\}$ and a set of attributes $\Phi = \{\phi_1, \phi_2, \ldots\}$. The percepts from $\Pi$ correspond to evidence which is assumed to originate from one object in the scene, and it corresponds to the concept of evidence presented in this chapter, which can either be segmented regions, or groups of smaller evidence grouped using a model-based approach. The attributes from $\Phi$ add together to form a feature vector of measured low-level properties describing the evidence in terms of colour, shape, texture, etc. Using the same notation as before, we can write

$$\Pi = E_{all}$$

$$\Phi = E = \{E_1, E_2, \ldots E_n\}.$$

The symbol system $\Sigma$ is represented by a set of individual symbols $\mathcal{X} = \{x_1, x_2, \ldots\}$ and a set of predicates $\mathcal{P} = \{p_1, p_2, \ldots\}$. In our middle-level framework, the symbols from $\mathcal{X}$ represent the instances of views from the high-level ontology which appear in a model, and can therefore be written as

$$\mathcal{X} = V_{all} = \{V_1, V_2, \ldots V_m\}.$$

These predicates are valuable for planning robot actions and tracking in complex temporal scenes, but are not as important for finding the best interpretation of a given image. The contextual classification framework presented in Section 3.2 assumes a class-posterior independence between low-level appearance and structural context used in the high level, so appearance-based context is not used for interpreting static images in this work. As a result, the appearance predicates are not a part of the described architecture, and play no role in matching at this moment. An extension of the middle-layer architecture to pass such predicates to the high level to improve the resulting high-level description of the scene is straight-forward.

Finally, the set of perceptual signatures $\Gamma = \{\gamma_1, \gamma_2, \ldots\}$ maps a combination of unary predicates from $\sigma \in 2^{\mathcal{P}}$ to an expected set of low-level values from $\Phi$, and is used to pass appearance expectations to the low level.

The previous section described the incremental evidence integration algorithm. If a high-level view $V_a$ with a perceptual signature $\gamma_c$ is successfully matched to evidence $E_b$, an anchor is defined using Coradeshi's formalism as

$$\alpha \leftarrow \langle V_a, E_b, \gamma_c \rangle \tag{3.31}$$

The classification and role assignment based on logical predicates acting on value ranges is error-prone in complex scene-interpretation tasks. Because of this, the class probabilities $P(V|E)$, contextual priors $P'(C)$ and spatial constraints are used to determine best matches in our framework, as described in Section 3.2, so $\gamma_c$ is omitted and replaced by a matching quality:

$$\alpha_1 \leftarrow \langle V_a, E_b, P(V_a|E_b, \text{context})\rangle \tag{3.32}$$

An anchor $\alpha$ describes a link between some evidence in the image and a high-level object which produced this evidence, and represents the strength of the link through the classification confidence. Such anchors remain until the whole scene is interpreted, and can affect the assignments of further evidence by revealing potential conflicts with already existing assignments, as described in Section 3.5.2.

The concept of anchoring can be naturally extended to temporal scenes, where anchors need to be maintained for periods of time. In this case, the position and size of the view, as well as the perceptual description is used to maintain these correspondences. Since this work deals with stepwise interpretation of static images, this aspect is not explored in more depth here.

## 3.8 Summary

In this chapter, a middle-layer architecture for an interpretation system was presented. There were three main contributions.

Since scene interpretation is generally performed as a stepwise process, a classification equation was presented, which allows for stepwise integration of evidence into an evolving interpretation. Given certain independence assumptions, updated *contextual* class priors can be integrated, allowing for improved classification as a result of evolving scene context.

It was demonstrated how calibrated conditional probabilities $P(C|E)$ can be obtained from a wide range of state-of-the-art algorithms, allowing them to profit from improved class priors. Additionally, the size and position uncertainty of the views was modelled probabilistically in order to pass such uncertainty to the high level.

Finally, the interface between the middle layer and the high-level system was defined, with the help of *views*. The requirements for a high-level system were defined for both probabilistic high-level systems, which provide location-dependent class priors, and crisp systems, which provide view hypotheses restricted by bounding boxes.

The following chapter focuses on the technical part of implementing this framework.

# Chapter 4

# The Matchbox

The architecture described in the previous chapter was implemented as a stand-alone middleware component which can connect to a number of high-level reasoning systems and a number of low-level detectors. This makes it possible to evaluate the performance of the proposed approach within a complete interpretation system and helps to demonstrate its feasibility. Since the task of the middle layer within an interpretation system is matching instances of high-level concepts to low-level image evidence, this software component is referred to as the “Matchbox”.

The Matchbox was implemented as a stand-alone program in C++, which uses remote procedure calls and standard file formats to communicate with other parts of the interpretation system. The advantage of this approach is that it can be easily mixed with software components written in other languages, as long as a simple interface is added to them. This chapter gives a short overview of its functionality and deals with some of the implemenation decisions..

## 4.1 Internal Architecture

This section describes the data structures used inside the Matchbox to represent the concepts introduced in Chapter 3. The descriptions closely follow the implementation, but some information was left out for clarity. For example, class properties are shown as public members, while the implementation used object-oriented encapsulation, where class members are accessed through *getter* and *setter* methods.

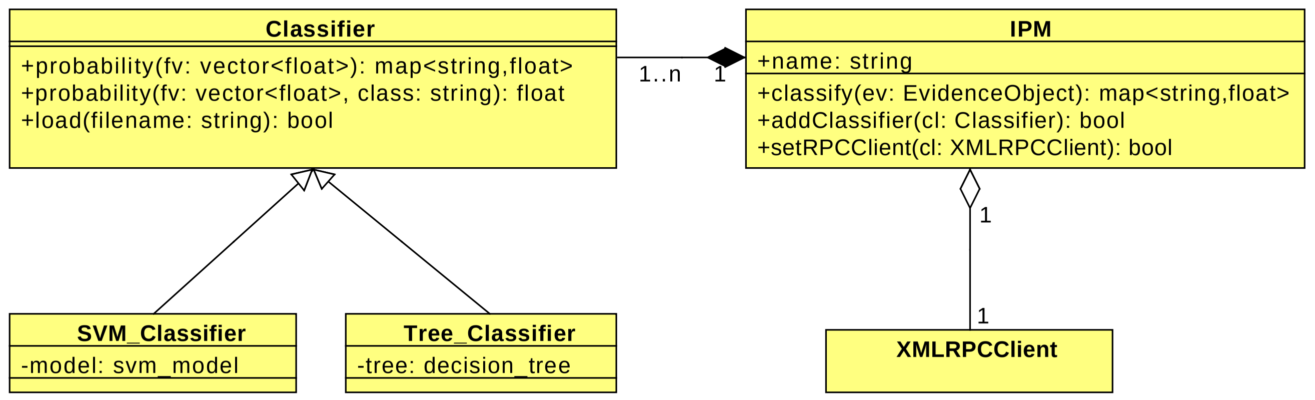

Classifier
+probability(fv: vector<float>): map<string,float>
+probability(fv: vector<float>, class: string): float
+load(filename: string): bool
IPM
+name: string
+classify(ev: EvidenceObject): map<string,float>
+addClassifier(cl: Classifier): bool
+setRPCClient(cl: XMLRPCClient): bool
1..n
1
1
1
SVM_Classifier
-model: svm_model
Tree_Classifier
-tree: decision_tree
XMLRPCClient

FIGURE 4.1: The class diagram showing the relation between the IPM class and the classifiers. See the text for an explanation of the class members and methods.

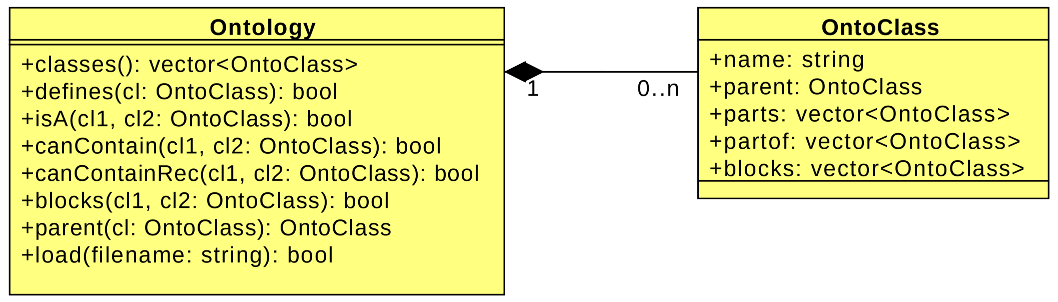

Ontology
+classes(): vector<OntoClass>
+defines(cl: OntoClass): bool
+isA(cl1, cl2: OntoClass): bool
+canContain(cl1, cl2: OntoClass): bool
+canContainRec(cl1, cl2: OntoClass): bool
+blocks(cl1, cl2: OntoClass): bool
+parent(cl: OntoClass): OntoClass
+load(filename: string): bool
1
0..n
OntoClass
+name: string
+parent: OntoClass
+parts: vector<OntoClass>
+partof: vector<OntoClass>
+blocks: vector<OntoClass>

FIGURE 4.2: The class diagram showing the data structure holding the ontology. The member functions provide information such as whether instances can have parts, the classes of the parts, and the specialisation relations.

### 4.1.1 Representing Image Processing Modules

An Image Processing Module (IPM) is a low-level algorithm which detects interesting phenomena in the image. They deliver a region (described by a bounding box and a bounding polygon), a feature vector, and sometimes a class. The access to IPMs is handled through the IPM class (see Figure 4.1), which contains a classifier used to provide class posteriors for the given evidence vector. In other words, each IPM knows what the delivered evidence "means" in terms of class probabilities. Additionally, the IPM class contains an instance of an RPC client class, which is used to pass RPC messages to the detector, e.g. in order to process an image.

The IPM class can classify evidence provided by the corresponding IPM through the **classify** method. In the case where evidence is a region with a feature vector, the vector is simply passed to the internal classifier. In the case of structured evidence (such as a collection of image patches), the IPM class can have several classifiers and the **classify** method combines the individual probabilities into a final class probability vector.

### 4.1.2 Representing Classifiers

Any probabilistic multi-class classifier can be integrated into the Matchbox in order to estimate the class probabilities based on evidence features. It should be pointed out that these classifiers provide a class probability $P(C|E)$, and the final classification into high-level concepts is performed by the Matchbox after comparing eligible evidence, integrating the dynamic priors, and checking whether any applicable high-level constraints (e.g. size, position) are met. Each classifier is associated with an IPM, and trained on the features provided by the IPM.

The "Classifier" class defines the interface to the rest of the Matchbox. Individual implementations are inherited from it and encapsulate the classifier-specific details. So far, a decision-tree probabilistic classifier and a probabilistic multi-class SVM classifier were integrated in this way. Different IPMs can use different classifiers, and the Matchbox can call the unified interface without having to know about the internal details.

The class diagram of for the "Classifier" class can be seen in Figure 4.1. The following methods are offered by each classifier:

- **load** takes a filename. It loads a learned classifier. In general, the classifiers are trained in a separate step and loaded at the beginning of the interpretation process.
- **probability** takes a feature vector and optionally a class. If a class is provided, the method returns the class posterior given the feature vector, i.e. $P(C|E)$. If no class is provided, it returns a C++ map (dictionary) giving a probability for each of the classes used during learning.

The decision tree classifier was written from scratch for this purpose, see [Terzić and Neumann, 2009b]. LibSVM [Chang and Lin, 2001] was used for the SVM implementation.

### 4.1.3 Representing the Ontology

The Matchbox can load a knowledge base expressed in the OWL format (see Section 4.2.4 for more details). The part that is important for the middle layer operation consists of the ontology (list of modelled classes), taxonomy (specialisation hierarchy of concepts) and class partonomy (the number and classes of parts a concept can have as parts). Figure 4.2 shows the data structure holding the ontology.

The "Ontology" class provides a number of methods which answer common questions about the classes of the domain:

- **load** takes a filename as an argument. It loads the ontology saved in the provided file.
- **classes** returns a list of all the classes defined by the ontology.
- **defines** takes a class as an argument. It returns "true" if the ontology defines the class, "false" otherwise.
- **parent** takes a class as an argument. It returns the parent class of the provided argument.
- **isA** takes two classes as arguments. It returns "true" if the first argument is a specialisation of the second, i.e. if the second class is an ancestor of the first. In order to determine this, the specialisation hierarchy is traversed until the sought class is found or the root node reached.
- **canContain** takes two classes as arguments. It returns "true" if the first class can have instances of the second class as parts. **canContainRec** checks recursively in the partonomical hierarchy.
- **blocks** takes two classes as arguments. It returns "false" if one class can occlude the other, "true" if the two classes block each other. It is used for reasoning about occlusion, when evidence suggests a different class than expected by the high level. It is essentially the inverse of the "can-occlude" relation introduced in the last chapter.

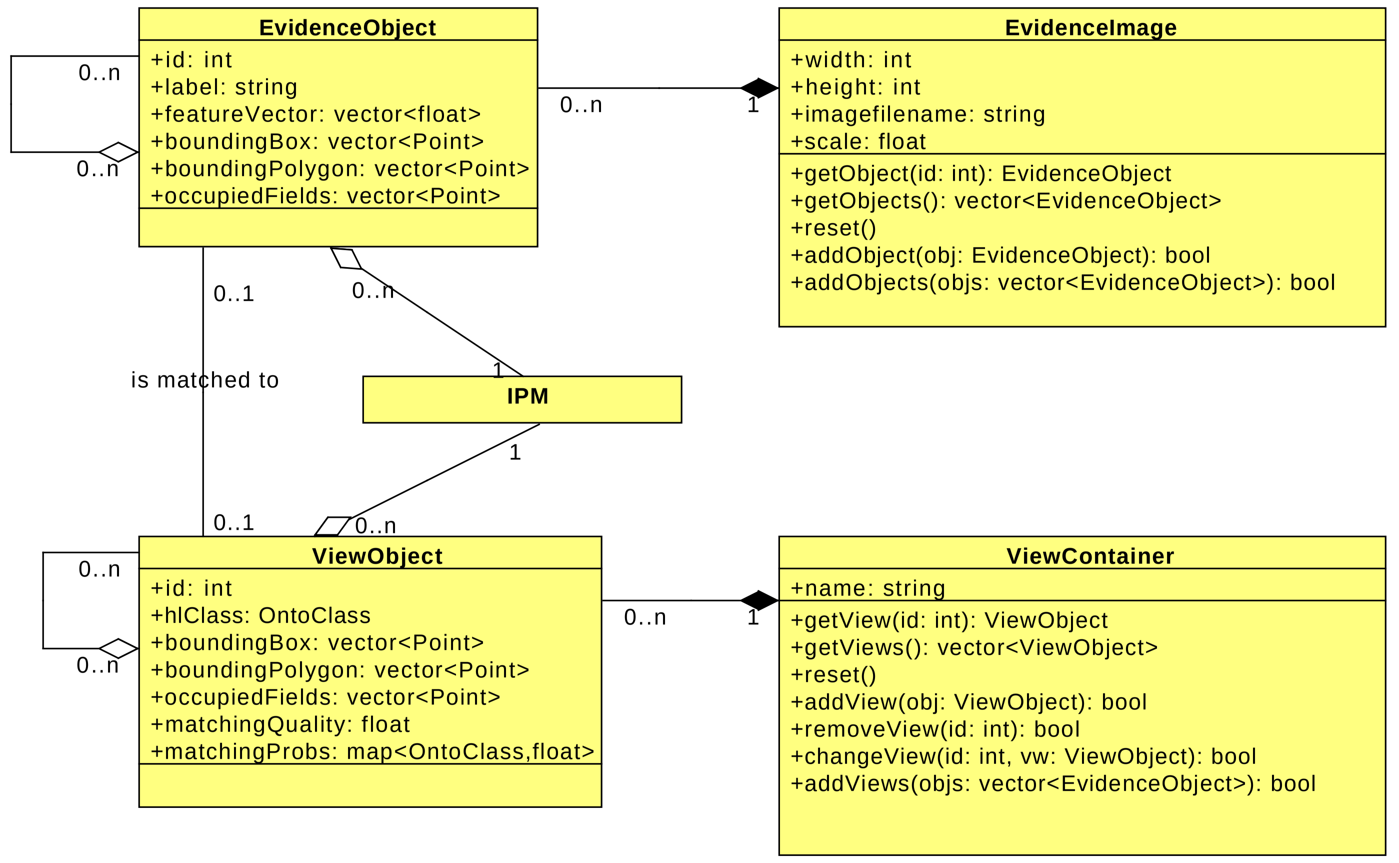

EvidenceObject
+id: int
+label: string
+featureVector: vector<float>
+boundingBox: vector<Point>
+boundingPolygon: vector<Point>
+occupiedFields: vector<Point>
EvidenceImage
+width: int
+height: int
+imagefilename: string
+scale: float
+getObject(id: int): EvidenceObject
+getObjects(): vector<EvidenceObject>
+reset()
+addObject(obj: EvidenceObject): bool
+addObjects(objs: vector<EvidenceObject>): bool
IPM
is matched to
ViewObject
+id: int
+hlClass: OntoClass
+boundingBox: vector<Point>
+boundingPolygon: vector<Point>
+occupiedFields: vector<Point>
+matchingQuality: float
+matchingProbs: map<OntoClass,float>
ViewContainer
+name: string
+getView(id: int): ViewObject
+getViews(): vector<ViewObject>
+reset()
+addView(obj: ViewObject): bool
+removeView(id: int): bool
+changeView(id: int, vw: ViewObject): bool
+addViews(objs: vector<EvidenceObject>): bool
0..n
1
0..1

Figure 4.3: The class diagram showing the data structures holding the evidence and views. The EvidenceObject and ViewObject instances are grouped together in an EvidenceImage and ViewContainer, respectively. Each piece of evidence and each matched view is associated with an IPM which produced the evidence.

### 4.1.4 Representing Evidence and Views

Each piece of evidence is stored in an instance of the class "EvidenceObject". It can represent a classified detection, a region, or a simple rectangular patch, such as delivered by a SIFT or Harris detector. A number of EvidenceObjects can be grouped together to form a compound EvidenceObject. This happens with structured evidence, like described in Section 5.2.2, or with hierarchical patch-based models such as the one presented in Section 3.3.3. The structure of the EvidenceObject class is shown in Figure 4.3.

- `label` is the class given by a specialised detector. If the detector does not classify, it is empty.
- `occupiedFields` is used for indexing, and explained in detail in Section 4.3.1.
- `FeatureVector` is a vector of floating point values describing the detection.

The remaining members are self explanatory.

A "ViewObject" representing a high-level view has a similar format.

- `matchingProbs` is a dictionary holding a class posterior for each of the possible classes.
- `hlClass` is the winning class, which is represented by one of the classes from the ontology (see Section 4.1.3).
- `matchingQuality` holds the probability that the view is an instance of `hlClass`. This information is useful for probabilistic reasoning systems, where the classification certainty can be used to influence further decisions.

All evidence is contained in a single container, "EvidenceImage", which holds basic information about the image being interpreted, and provides $\mathcal{O}(1)$ access to all elements based on the ID. The views are grouped together in a similar structure, called "ViewContainer". In addition to the methods providece by EvidenceImage, ViewContainer also allows easy adding, removing and changing of views. This is necessary because views can be changed in the high level based on high-level constraints, and such changes need to be mirrored inside the Matchbox. The RPC interface for this tight high-level and middle-layer integration is described in Section 4.2.1.

### 4.1.5 Representing Correspondences

Once an EvidenceObject is matched to a ViewObject, the Matchbox represents this relation internally in a special class. The matches are represented as triples containing Evidence, View and matching quality, as described in Section 3.7.1. In addition to tracking the corresponding pairs of views and evidence, the MatchingTable class can also remove a correspondence and prohibit the matching of certain evidence-view combinations (in case the matching is revoked due to a conflict). Every time a new match is made by the Matchbox, it first checks whether the attempted match is prohibited, to avoid repeating faulty assignments.

## 4.2 Input and Output

The Matchbox is a stand-alone component that communicates with other parts of the interpretation system. The primary communication channel is the RPC interface which allows passing messages and data between the individual components in real time. Additionally, the Matchbox can load and save several file formats. This makes it possible to save the current state of the interpretation and to test individual parts of the system. The following subsections describe the implemented input and output facilities.

### 4.2.1 Remote Procedure Calls

The Matchbox uses an XMLRPC interface to communicate with other parts of the interpretation system. There are many RPC mechanisms available for these types of tasks, including CORBA, SOAP and DCOM, but many of them are very complex and require specialist knowledge. The advantages of XMLRPC are that it is lightweight, portable, and simple to implement. Since the Matchbox is designed to interface easily with a variety of existing interpretation systems and low-level algorithms, this ease of development is a very important consideration.

The Matchbox operates both as an RPC client (calling procedures offered by other processes) and an RPC server (providing services to other processes). This section briefly introduces the services offered by the Matchbox and the services it expects from other components of the interpretation system. In addition to being able to call services from other processes, the XMLRPC interface is very important for synchronising the data between the Matchbox and the high-level modules. Since high-level views can be affected both by high-level constraints and by low-level detections, it is important to work on the same data. RPC allows the different modules to share their changes with each other and keep the data consistent. Figure 4.4 shows the basic blocks of the interpretation system and the RPC communication channels, described in more details below.

**Data structures** provided by the XMLRPC framework are used to pass complex datatypes between modules. The most important structure is an XMLRPC struct representing a high-level view. It consists of the following elements:

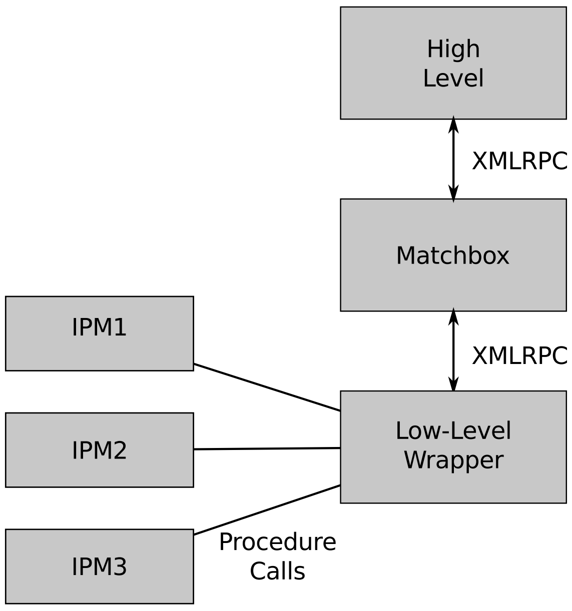


Figure 4.4: The XMLRPC interface between the main blocks of the interpretation system. The boxes represent independent processes. The Matchbox communicates with the high level and the low-level wrapper through an XMLRPC interface. The low-level wrapper calls the individual detectors by starting a detector every time it is requested. The RPC interface only needs to be programmed once, avoiding code duplication.

- `Pos-X-1`, `Pos-X-2`, `Pos-Y-1` and `Pos-Y-2` are either floating point values or intervals (see below). They describe the upper-left and lower-right corner of the bounding box.
- `Size-X` and `Size-Y` are either floating point values or intervals. They describe the horizontal and vertical size of the view.
- `MatchingProbabilities` is a dictionary which maps strings (corresponding to domain classes) to floating numbers representing a posteriory probabilities. It is used to pass the classifications to the high-level.
- `type` is a string identifying the strongest class (for non-probabilistic reasoning modules).
- `MatchingQuality` is a floating-point value indicating the probability for the strongest class (for non-probabilistic reasoning modules).
- `id` is a unique integer id identifying the view.
- `ipm` is a string indicating which image processing module produced the detection.

Some of the values can also be interval valued. In this case the elements like `Pos-X-1` and `Size-X` are not real-valued, but are structs themselves, with two elements:

- `min` is a floating-point value indicating the lower bound of the interval.

- `max` is a floating-point value indicating the upper bound of the interval.

The Matchbox only works with convex intervals, i.e. there is always exactly one lower bound and exactly one upper bound. Interval arithmetic allowing "holes" within the intervals is not supported. The rest of this section describes the XMLRPC procedures implemented in the Matchbox.

**Client side** procedures are those which the Matchbox needs to call during an interpretation process, and which must be provided by other modules. Some of these deal with communicating the current state of the interpretation to the high-level process. The RPC methods which need to be implemented by the high level are described first.

- `ensureViewHL` takes an XMLRPC struct representing a view (described above). It communicates the current state of views to the high level, including the class, position, bounding box, etc.
- `deleteViewHL` takes an integer ID. It tells the high level to delete it.
- `getPriorsHL` takes a vector of floating-point values describing a bounding box. It asks the (probabilistic) high-level system to return a list of prior probabilities for all the classes for this region.

The low-level detectors can also be controlled via the XMLRPC interface. Unlike high-level, they do not need a tight integration with the middle layer, as they are usually called with a set of parameters, deliver the detections, and terminate. Instead of making an RPC-aware process out of each detector, a simple C++ wrapper was implemented. The wrapper is a separate process which provides the XMLRPC interface for the Matchbox and calls the required modules as needed. This reduces the complexity of the RPC interface, but also makes it easy to control detectors written in tools such as Matlab. The low-level wrapper offers a number of services for controlling the detectors:

- `init` initializes the detector, running any necessary setup code.
- `processImage` takes an image filename, runs the detector on the image, and returns the detections in the XML format (see Section 4.2.2 for details).
- `processImageROI` takes a filename, a vector of floating-point values describing a bounding box, and (optionally) a sensitivity parameter represented by a floating point value in the range $[0, 1]$. It returns all the detections in the region, using the specified sensitivity.

**Server side** procedures are offered by the Matchbox and implement the services described in Chapter 3. They can be called by other modules during an interpretation. The important procedures are described below.

- `reset` takes no arguments. It initialises the Matchbox by deleting all data and resetting all values.
- `ensureView` takes an integer ID and an XML struct representing the view. It updates the internal Matchbox description of the view to reflect the supplied data.
- `deleteView` takes an integer ID. It deletes the view with the supplied ID, as it no longer exists in high level.
- `loadKnowledgeBase` takes a path to a file. It loads the ontology used for the interpretation from the specified OWL file. See Section 4.2.4 for more information.
- `loadIPMDetections` takes three string arguments. The first is the file where low-level detections are saved (see 4.2.2 for a description of the file format). The second one is the file describing the classifier used for this detector. The third one is a string identifier holding the name of the detector. The procedure loads the detections and the corresponding classifier into the Matchbox, allowing it to produce probability estimates.
- `loadFilterOnto` and `loadInstanceFilter` both take a path to a file specifying a desired filter. See Section 4.4.1 for a description of the filtering facilities.
- `createViewsAboveConfidence` takes a floating-point value in the range $[0, 1]$. It creates high-level views from evidence for all views where the classification confidence (defined in Section 3.2) is above the given threshold. This is the bottom-up classification step.
- `matchHypothesisToEvidence` takes an XMLRPC struct representing a view. It attempts to match the provided view to the available low-level evidence. It returns a modified view, with the size and the bounding box updated to reflect the matched evidence. If the view cannot be matched, -1 is returned as matching quality.
- `unmatchedEvidence` takes no arguments. It returns the amount of evidence which still hasn't been explained by high-level views (is left unclassified). It can be used for constructing interpretation loops.

A number of additional services which are not crucial for the basic interpretation process have been left out for clarity. These include auxiliary functionality like saving the detections, saving the high-level views, writing out matching statistics, etc.

### 4.2.2 XML Evidence Description

The evidence passed by low-level detectors has the same format as the annotations. Each object is described by a bounding polygon. Additionally, the compositional hierarchy with the has-part relationships can be represented in the case of annotated images or

TABLE 4.1: XML File Structure

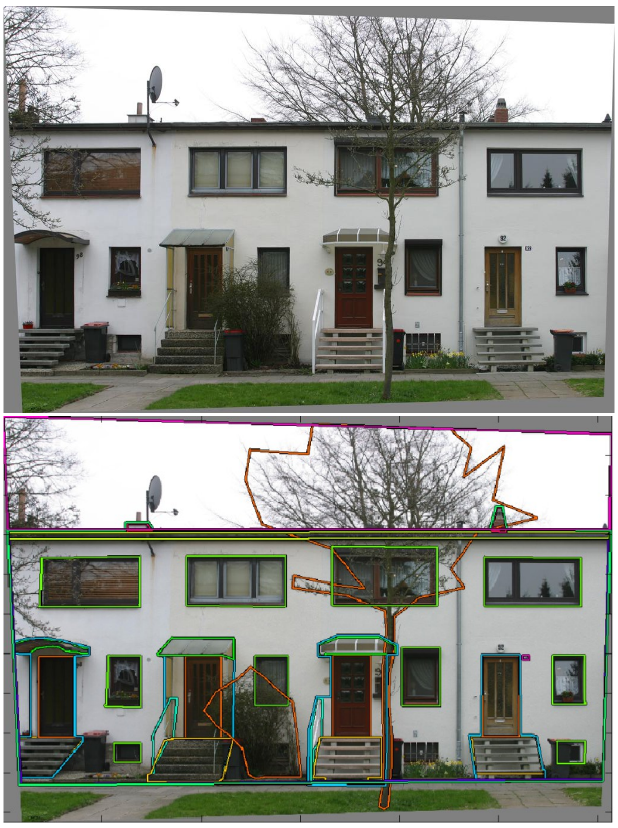
| | |
|---|---|
| `annotation` | |
|   `filename` | the filename of the corresponding image file |
|   `folder` | the folder name of the corresponding image file |
|   `sourceAnnotationXML` | the version string of the annotation tool |
|   `rectified` | 1 := *image is rectified* and 0:= *image is not rectified* |
|   `imageWidth` | the width of the corresponding image |
|   `imageHeight` | the height of the corresponding image |
|   `transformationMatrix` | the transformation matrix used for the rectification |
|   `annotatedClasses` | |
|     `className` | |
|     ⋮ | |
|   `scale` | estimate of the size of *pixel per cm* |
|   `object` | an object in the scene, this can be an aggregate or a primitive |
|     `name` | the type of the object, e.g. `window` |
|     `objectID` | an ID for the object (unique in the complete data set) |
|     `date` | the date of the annotation |
|     `sourceAnnotation` | the person that annotated the image |
|     `polygon` | the polygon that describes the object |
|       `pt` | a point in the polygon |
|         `x` | the x coordinate of the point |
|         `y` | the y coordinate of the point |
|       ⋮ | |
|     `objectParts` | a list of `objectID`s of the parts that belong to this object |
|     `confidence` | a measure of confidence, if the object was automatically retrieved from the image |

detectors which can detect structured evidence (e.g. detectors which use a grammar to only detect aligned regions). The annotations are described in an XML format which is an extension of the LabelMe XML format. The LabelMe XML format also defines objects as bounding polygons, but does not include partonomical relations.

All the XML tags and the structure of an XML file are shown in Table 4.1, where (⋮) indicates that multiple entities may be present. The indents represent *has-part* relationships, while tags on the same level, in the same scope, are siblings. A partial tree structure of the compositional hierarchy of an image is shown in Figure 4.5. The new tags compared to the LabelMe XML format are:

- `imageWidth` and `imageHeight`, which contain the size of the original image,
- `scale`, which is an estimate of how many pixels represent a centimeter at the facade level,
- `rectified`, which indicates whether the image has been rectified, and `transformationMatrix`, which contains the matrix used for the rectification

- `annotatedClasses`, which contains a list of classes that are annotated in the image, each described by a `className` tag. In the case of a specialised detector, this lists the classes which the detector is trained to detect,
- `objectID`, which is an object identifier string, and
- `objectParts`, which is a comma-separated list of the objectIDs of all the parts belonging to an object.



```
single family home
  ...
single family home
  ...
single family home
  roof
    chimney
  facade
    upper floor
      window
    ground floor
      entrance
        door
        railing
        stairs
        canopy
        railing
    window
    window
single family home
  ...
vegetation
vegetation
sky
ground
```

FIGURE 4.5: Example image with the corresponding annotation shown below. The *has-part* relationships are shown on the right, with the partonomy indicated by indentation levels. Objects with the same scope and indentation level are siblings.

The Matchbox has an internal XML parser based on the *expat* library for loading and saving data in this format.

### 4.2.3 RDF Interpretation Format

In addition to saving the detected evidence, the Matchbox can also load and save a partial interpretation of a scene. This makes it possible to resume an interpretation, or save the current state of an interpretation in order to visualise the result using an external viewer. The partial interpretation is represented as a set of RDF triples having the Subject-Predicate-Object form. Since the *Raptor* library is used for loading, any supported RDF serialisation format is supported, though in practice only the XML serialisation was used.

The classes of the objects modelled in a particular domain have to be defined in an OWL ontology (described in the following section). The RDF handling facilities were used in the context of interpreting building façades and used the eTRIMS ontology of building parts. An example of a view from the building domain in this format is shown in Figure 4.6.

In addition to the object classes, a number of predicates are defined, whose objects represent parameters of the views of scene objects:

- *Size-X* and *Size-Y*, specifying the width and height of the view of an object in the image,
- *Pos-X-1* and *Pos-Y-1*, specifying the x and y coordinates of the upper-left corner of the bounding box,
- *Pos-X-2* and *Pos-Y-2*, specifying the lower right corner of the bounding box,
- *id*, specifying the unique ID of the view,

Additionally, the class *Interval* is defined, which makes it possible to express value ranges. For example, if the object of a *Size-X* predicate is not a number, but an *Interval* object, this means that there is uncertainty about the exact size of a view, and that there is a range of allowed values. The range is defined through two additional predicates:

- *upper*, specifying the upper bound, and
- *lower*, specifying the lower bound.

The RDF format lends itself well to representing ontologies and relations between objects, and there are tools which make saving from LISP-based representations such as SWCLOS easy. This makes it a good fit for logic-based interpretation systems such as SCENIC and SCENIOR, which can save results in this format with little extra work required. However, due to its complexity, it is generally easier and more effective to use the XMLRPC interface for all communication, and only use the RDF format for visualising interpretations.

### 4.2.4 OWL Knowledge Base

Scene interpretation systems work with an ontology of concepts. In order to supply valid input to the high-level reasoning system, the Matchbox can load ontologies in the Web

```
<rdf:RDF xmlns:gx="http://www.galaxy-express.co.jp/SW/SWCLOS#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:eo="http://etrims.home.dyndns.org/eOnto#" >

  <eo:Roof-View rdf:about=
        "http://etrims.home.dyndns.org/eOnto#Roof-View-77" >
    <eo:HighLevelName>Roof-View-16</eo:HighLevelName>
    <eo:id>77</eo:id>
    <eo:Size-Y>347</eo:Size-Y>
    <eo:Size-X>625</eo:Size-X>
    <eo:Pos-X-2>
        <eo:Interval>
            <eo:lower>700</eo:lower>
            <eo:upper>747</eo:upper>
        </eo:Interval>
    </eo:Pos-X-2>
    <eo:Pos-X-1>
        <eo:Interval>
            <eo:lower>100</eo:lower>
            <eo:upper>115</eo:upper>
        </eo:Interval>
    </eo:Pos-X-1>
    <eo:Pos-Y-1>586</eo:Pos-Y-1>
    <eo:Pos-Y-2>934</eo:Pos-Y-2>
    <eo:Label>Roof-Label</eo:Label>
    <eo:observation-of rdf:resource=
        "http://etrims.home.dyndns.org/eOnto#Facade-Scene-10016" />
    <eo:view-of rdf:resource=
        "http://etrims.home.dyndns.org/eOnto#Roof-158" />
  </eo:Roof-View>

</rdf:RDF>
```

FIGURE 4.6: Example view saved as serialised RDF by the SCENIC interpretation system, which can be loaded by the Matchbox. Several of the predicates, like "observation-of" and "HighLevelName" relate to internal SCENIC attributes, and are ignored by the Matchbox. The "eo" prefix is a shorthand for the eTRIMS ontology file holding the definitions.

Ontology Language (OWL). Since OWL is a standard ontology language, it is easier to adapt to new domains and ontologies.

As described in the previous chapter, two relations are important for the classification task. The taxonomical specialisation (expressed in OWL either as a subclass or an intersection with an existing class) and the partonomy (expressed through the newly defined properties "has-elements" and "element-of"). Figure 4.7 shows a truncated excerpt from the façade domain, where the class "Balcony" is defined as a specialisation chain from the root concept "eThing" and is given a list of possible parts (in this case, exactly one “Door” object). The complete ontology used for interpreting façade scenes was more complex, but too large to show here.

## 4.3 Indexing

Dealing with evidence from many different detectors can lead to a large number of evidence objects to be processed at any matching step, especially if interest points are used. In simpler problems, iterating through all the detected evidence is not a big problem, but the complexity increases considerably in dynamic scenes involving motion. With this in mind, Matchbox uses indexing internally to provide fast access to evidence and views based on important criteria such as position or type.

### 4.3.1 Spatial Maps

Scene interpretation helps to focus the attention on a small part of the image, give updated prior probabilities for small regions of interest, or make hypotheses for objects in certain locations. In order to quickly access the low-level evidence in the interesting regions, it is useful to have a spatial indexing scheme, and thus avoid checking each evidence individually every time a matching request is made.

In order to speed up the access to the evidence in specific areas of the image, the image is divided into rectangular blocks (the Matchbox uses $100 \times 100$ at the moment) and each block is associated with an `ID list` identifying the objects whose bounding polygons overlap with the block. Additionally, each piece of evidence carries a `block list` of all the spatial blocks it overlaps with. The same is repeated with the high-level views, giving two 2D maps, which are represented by 2D arrays of lists. This allows for a number of fast operations which are commonly used:

- **Checking for polygon intersection** consists of checking the block lists of the two objects and seeing if there are common blocks between the two.
- **Finding evidence within a bounding box** consists of finding all blocks within the bounding box and merging the corresponding lists of IDs. At the end, only the small number of remaining evidence polygons need to be checked for real overlap (e.g. by using polygon clipping).

Since the spatial maps used for evidence and views have the same dimensions, finding all evidence that might confirm or refute a high-level view is trivial, and avoids trying all the evidence in the image. This is especially important when using very complex models with thousands of parts and when using multiple low-level detectors.

```
<rdf:RDF xmlns:gx="http://www.galaxy-express.co.jp/SW/SWCLOS#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:eo="http://etrims.home.dyndns.org/eOnto#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:owl="http://www.w3.org/2002/07/owl#" >

  <owl:Class rdf:about="http://etrims.home.dyndns.org/eOnto#eThing" />

  <owl:Class rdf:about=
        "http://etrims.home.dyndns.org/eOnto#Real-World-Entity" >
    <rdfs:subClassOf rdf:resource=
        "http://etrims.home.dyndns.org/eOnto#eThing" />
  </owl:Class>

  <owl:Class rdf:about=
        "http://etrims.home.dyndns.org/eOnto#Scene-Aggregate" >
    <rdfs:subClassOf rdf:resource=
        "http://etrims.home.dyndns.org/eOnto#Real-World-Entity" />
  </owl:Class>

  <rdf:Property rdf:about=
        "http://etrims.home.dyndns.org/eOnto#has-elements" >
    <rdfs:domain rdf:resource=
        "http://etrims.home.dyndns.org/eOnto#Scene-Aggregate" />
    <rdfs:range rdf:resource=
        "http://etrims.home.dyndns.org/eOnto#Real-World-Entity" />
  </rdf:Property>

  <owl:Class rdf:about="http://etrims.home.dyndns.org/eOnto#Balcony" >
    <owl:intersectionOf rdf:resource=
         "http://etrims.home.dyndns.org/eOnto#Scene-Aggregate" />
    <owl:intersectionOf>
      <owl:allValuesFromRestriction>
        <owl:onProperty rdf:resource=
            "http://etrims.home.dyndns.org/eOnto#has-elements" />
        <owl:minCardinality>1</owl:minCardinality>
        <owl:maxCardinality>1</owl:maxCardinality>
        <owl:allValuesFrom rdf:resource=
            "http://etrims.home.dyndns.org/eOnto#Door" />
      </owl:allValuesFromRestriction>
    </owl:intersectionOf>
  </owl:Class>

</rdf:RDF>
```

Figure 4.7: An example ontology from the façade domain in the OWL format. All the objects in the scene are specialisation of the root node "eThing". Property "has-elements" is defined to express possible partonomical relations between classes. Class "Balcony" is defined as a specialisation of the "Scene-Aggregate" class with the "has-elements" property restricted to allow exactly one "Door" as a part.

In summary, the 2D maps used inside the Matchbox provide easy testing of overlaps, quick ROI focussing, and are biologically inspired.

### 4.3.2 Efficient Data Handling

Complex intepretation systems can deal with a number of low-level detectors which produce a large amount of evidence. As an example, the Stable Region Detector of [Drauschke, 2009] can produce over a thousand regions in an image of average size. Patch-based detectors regularly produce hundreds to thousands of interest points, which can be grouped arbitrarily. Large high-level ontologies can also be composed of many view classes, which results in a large number of potential assignments at each decision step. In order to deal with this complexity, several indexing methods are used inside the Matchbox to speed up access to data.

**ID access:** Each view and each piece of evidence is characterised by a unique number, called ID. All evidence and views are organised in two hash tables which allow access in constant time given the ID.

**Type access:** A list of all views of a certain class can be obtained in logarithmic time, which was fast enough for all ontologies used so far.

**IPM access:** All evidence produced by a specific detector can be grouped together and accessed in constant time.

**Probability caching:** Some classification tasks involve a large number of probabilities which have to be multiplied together in order to produce a class probability list, for example patch-based detector described in Section 3.3.3. Depending on the classifier used and the amount of data, this can take a second or more to calculate for all classes. Such results are cached for later use, and to only select a relevant subset of classes to calculate the updated posterior at later steps.

The described facilities, while straightforward, help to keep the evidence and views managable and efficient during regular operation.

## 4.4 Auxilliary Facilities

In addition to the main services offered by the Matchbox, there are several additional tools which can be helpful for setting up experiments and interpreting scenes. They are described in short in this section.

### 4.4.1 Filtering

Under normal circumstances, a middle layer classifies detected evidence into instances of concepts from the domain ontology, maximising the classification confidence. However, there are cases where it is required to restrict this behaviour for some evidence, or for a set of classes. There are three common scenarios:

- **semi-automatic segmentation**, where a part of the evidence is supplied from the annotation and the rest of the image is interpreted by the reasoning system,
- **evaluating high-level performance**, where the bottom-up creation of some classes is suppressed in order to assess how well the high-level can predict them, and
- **interactive interpretation**, where a user can interactively specify which of the evidence should be classified (e.g. through an external GUI) and check how it affects the interpretation process.

There are two filtering mechanisms integrated into the Matchbox for this purpose:

- **Instance filter**, which receives a list of IDs from the XMLRPC interface or loads them from a text file and ignores such evidence during classification, and
- **Class filter**, which loads a set of classes, either as a list of labels or as an additional *filter ontology* in the OWL format, and does not create instances of such classes during the bottom-up steps.

The filtering facilities make it easier to use the Matchbox for evaluating the performance of different parts of the interpretation system. They were used for debugging the interpretations provided by the SCENIC system and the Bayesian Compositional Hierarchies.

## 4.5 Working Interfaces

In order to demonstrate the generic nature of the proposed architecture, a number of interfaces to both high-level and low-level modules have been written. They are listed here for completeness.

**Crisp Reasoning Systems** produce an interpretation based on crisp constraints operating on instances of classes from an ontology, and can provide hypotheses in terms of expected positions, value ranges and allowed classes. The integration was performed with

- the *SCENIC* reasoning system [Hotz and Neumann, 2005], written in LISP and based on the configuration tool KONWERK.

**Probabilistic Reasoning Systems** treat object instances as probabilistic variables and estimate the prior probabilities for location, size and class of other objects in the scene based on the observed evidence. These priors can be used to improve the classification of remaining objects. The two systems integrated with the Matchbox are

- *Bayesian Compositional Hierarchies* [Neumann, 2008], an object-based extension of Bayesian Networks, and
- *Conditional Markov Random Fields* [Heesch and Petrou, 2009], which estimate the class priors for regions based on the classes of the surrounding regions.

In the case of low-level detectors, one can differentiatiate betwee three types: (i) classifying detectors, (ii) segmenters, and (iii) patch-based hierarchical detectors.

**Classifying Detectors** specialise in detecting objects of a certain class only. The detectors used were

- *Annotation loader*, which loads the hand-annotated regions and labels, and can be used both a ground-truth or to initialise a partial interpretation with correct evidence,
- *Trainable AdaBoost detector* [Šochman and Matas, 2009], which can be trained for a number of classes, and
- *Visual structure detector* [Terzić et al., 2010], which is a combination of the AdaBoost detector and a grammar imposing a structure on the detected entities.

**Segmenters** detect regions which could correspond to objects, but do not classify them. In order to provide a classification probability, features are extracted from the detected regions and probabilities are estimated using a probabilistic classifier. The segmenters used were

- *Interest region detector* [Jahangiri and Petrou, 2008], which detects salient regions (blobs) in the image, and
- *Stable Region Detector* [Drauschke, 2009], which finds regions which are stable across several scales.

**Patch-based Detectors** include methods which detect objects based on representative image patches or compositions thereof. The following detector was implemented and integrated into the Matchbox:

- *Patch-based hierarchical model* of [Ommer and Buhmann, 2010], which models compositions of parts hierarchically, resulting in class probabilities.

Due to the modular and generic design of the Matchbox, only a few hundred lines of code were needed in each case. The added code provides the XMLRPC interface for inter-process message passing and (in the case of low-level modules) a simple parser for the LabelMe XML format. The remainder (integrating priors, modelling detection errors, finding the most suitable evidence, etc.) is handled by the Matchbox internals. Also interesting is the ability to integrate low-level detectors providing evidence at different levels: interest points, regions, or already classified detections.

This ease of integration of state of the art modules written in a variety of programming languages is new in the field of scene interpretation, where most interpretation systems are built from modules written specifically for a particular system and, often, for a particular application or domain.

## 4.6 Summary

This chapter described the implementation of a software component implementing the proposed architecture. It covered the internal representation of the different middle-level elements such as views, evidence and classifiers, bridging the gap between the theory and implementation. Data exchange between the middle layer and other parts of the interpretation system was explained in terms of the supported data formats and the RPC interface for realtime communication. Finally, some additional services such as filtering and indexing were described, which are useful for performance reasons and to help set up experiments.

The following chapter will present the experimental results of different combinations of these high-level and low-level modules.

# Chapter 5

# Evaluation

A number of experiments using the proposed middle-layer are presented in this chapter. Since the middle-layer is designed to be a part of an interpretation system, many experiments test a complete interpretation system consisting of low-level image processing modules and a high-level reasoning system. The experiments were obtained over a long period of time, during different research projects, so they showcase the integration with a wide range of image processing modules and high-level reasoning systems.

## 5.1 Description of Data

Standard benchmarks are very important for evaluating the results of new Computer Vision methods. Unfortunately, there are few datasets available which are suitable for evaluating entire scene interpretations. In many common annotated datasets used by the Computer Vision community, such as the TU Darmstadt Database [Leibe et al., 2004], UIUC Car Database [Agarwal et al., 2002], the VOC Challenge datasets [Everingham, 2006], and the Caltech 101 [Fergus and Perona, 2003], MIT-CSAIL [Torralba et al., 2003] and TU Graz [Opelt et al., 2006] datasets, only a small number of the objects in the scene are annotated.

The experiments in this domain were done on two datasets: Caltech 101 for structured appearance models, and the eTRIMS dataset of annotated façade images for classification of segmented objects and to test complete interpretations.

### 5.1.1 The Façade Domain

Much of the evaluation reported in this thesis was done on the eTRIMS Interpretation Dataset B, which consists of 200 fully annotated facade images [Hartz et al., 2010]. This domain is not only relevant due to the increasing interest in 3D city modelling, landmark

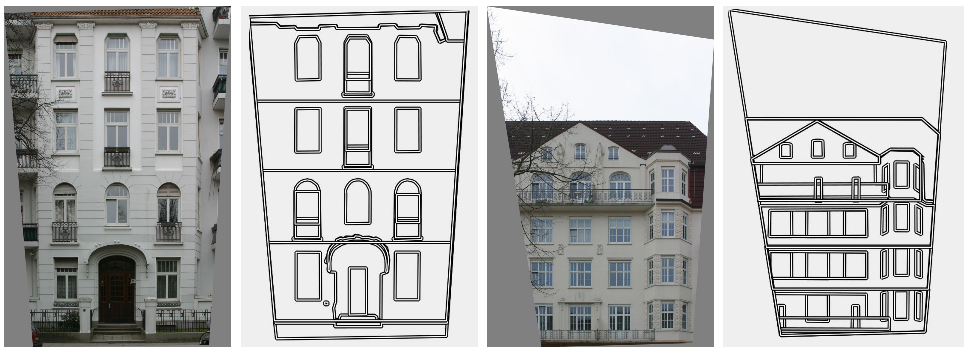


FIGURE 5.1: Two examples from the dataset. The image on the left is followed by the corresponding annotation on the right. The grey regions are the result of the image rectification process.

recognition and autonomous navigation, but also presents challenges which are difficult to solve at the object-level without the use of context:

- the appearance of many classes is extremely variable in terms of shape, colour and texture,
- objects of different classes are often visually similar, e.g. doors and windows,
- some of the classes, such as balconies and entrances, are loose configurations of parts and don't have a characteristic appearance by themselves.

Since the domain of building façades is a highly structured one, higher-level knowledge and image structure can be used to improve the performance, thus measuring the contribution of top-down processing steps on the overall scene understanding.

The dataset contains 200 images of buildings from various cities including: Basel (Switzerland), Berlin, Bonn, Hamburg, Heidelberg, Karlsruhe, Munich (Germany), Prague (Czech Republic) and some cities from the United Kingdom. For each image, an annotation is provided, marking all visible objects across multiple levels of hierarchy. The annotations are provided as bounding polygons which closely follow the objects' contours, but are not accurate at a pixel-level. The polygons are closed (the starting and ending points are the same). All of the images used in the dataset are *rectified*, to make it easier to exploit the horizontal and vertical alignments specific to the domain.

An XML format is used for representing the annotations. The format is an extension of the well-known MIT LabelMe [Russell et al., 2005] format, for which a number of tools are available, and which is easy to parse. The format was described in more detail in Section 4.2.2. The annotations can be viewed and modified using the Annotation Tool developed at University of Bonn [Korč and Schneider, 2007].

The full dataset consists of objects from 30 different classes. The classes *Canopy, Car, Chimney, Door, Ground, Pavement, Person, Railing, Road, Sign, Sky, Stairs, Vegetation* and *Window* represent simple objects without parts, and are referred to as *primitives*. They are annotated as bounding polygons with a class label. The remaining classes represent formations of other objects, which may or may not have a typical appearance on their own, and are referred to as *aggregates*. For example, a *Balcony* can contain doors, windows and railings, a *Window-Array* is a row of windows, and an *Upper-Floor* is a horizontal alignment of windows and balconies above the ground level. Such aggregate classes are annotated using bounding polygons, a class label, and a list of their parts (expressed by IDs which are unique in the database). The objects are annotated to the extent that they are visible, excluding the occluded parts or parts that are located outside the image.

## 5.2 Interpretation Using a Crisp High-Level System

This section describes the experiments where the Matchbox was used as a part of the SCENIC interpretation system. Detected objects were passed to the high-level system as classified views, and high-level predictions were passed down as view hypotheses to be confirmed or rejected.

### 5.2.1 Evaluating Incremental Model Learning

In this experiment, the Matchbox was used as a part of an evaluation framework testing the performance of automatically learned high-level aggregate concepts. The interpretation result was compared to a standard, correct interpretation. This is essentially a teacher giving the correct interpretation and evaluating the student's performance. In this case, the teacher's input is represented by an annotated image database. SCENIC interpretations are aggregate structures which describe the observed evidence, fulfilling the constraints imposed by the knowledge base. The annotations used in this experiment (see Section 5.1.1) include the partonomical relationships between objects, which mirror the compositional structure of a SCENIC interpretation, and can therefore act as the ground truth input by the teacher.

In order to evaluate the interpretation, it is important that the evaluation process stays as close as possible to the standard interpretation process. By comparing the interpretation result to the one provided by the teacher, it is possible to evaluate how well a certain version of SCENIC can interpret scenes in a given domain, given a certain knowledge base and a certain set of image processing modules. In this experiment, the output of the high-level learning module based on Version Space Learning [Hartz and Neumann, 2007] was tested.

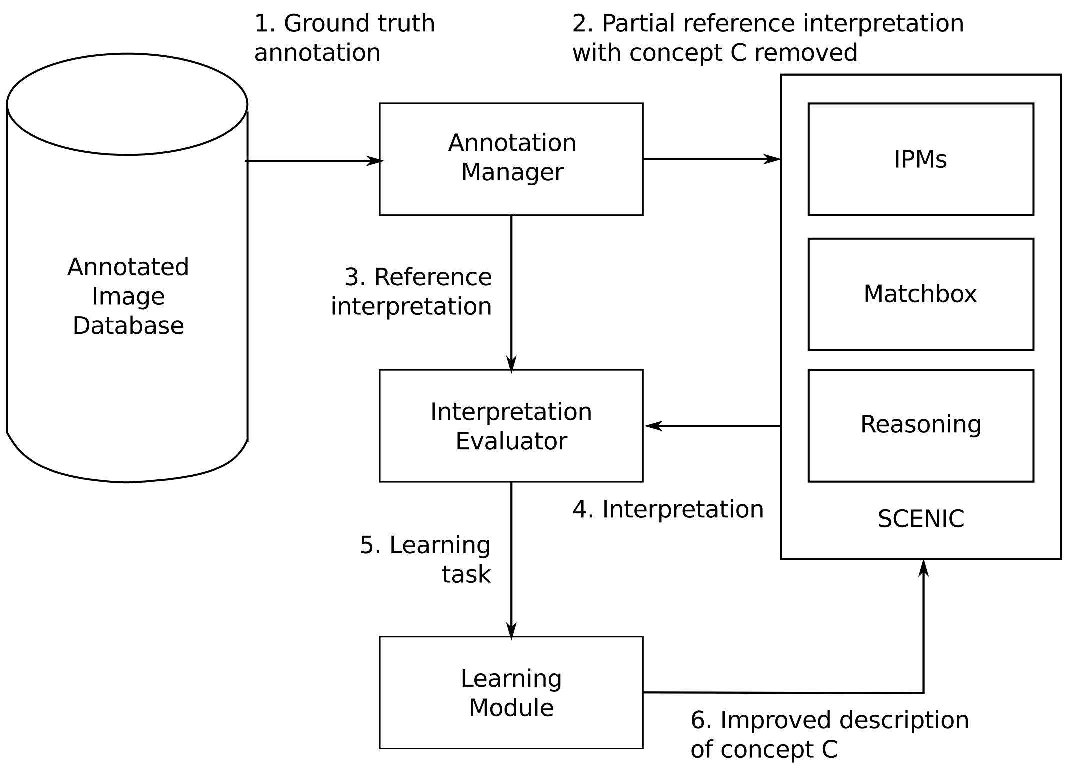

1. Ground truth annotation
2. Partial reference interpretation with concept C removed
Annotated Image Database
Annotation Manager
IPMs
Matchbox
Reasoning
SCENIC
3. Reference interpretation
Interpretation Evaluator
4. Interpretation
5. Learning task
Learning Module
6. Improved description of concept C

FIGURE 5.2: The SCENIC evaluation framework used for evaluating learned concepts. The Matchbox is a part of the interpretation system.

The task of the Matchbox in this case was not to classify ambiguous evidence, but to serve as an interface to the annotations representing the ground truth and to confirm or refute the high-level hypotheses by comparing them to the annotated regions. In this experiment, the Matchbox takes over the task of matching high-level views to annotations, allowing the high-level to operate exactly the same way as during any other interpretation. As a result, the high-level does not need to be adapted for evaluation purposes, removing a potential source of errors. Although the Matchbox does not play a central role in this experiment, it made the implementation of the evaluation loop easier.

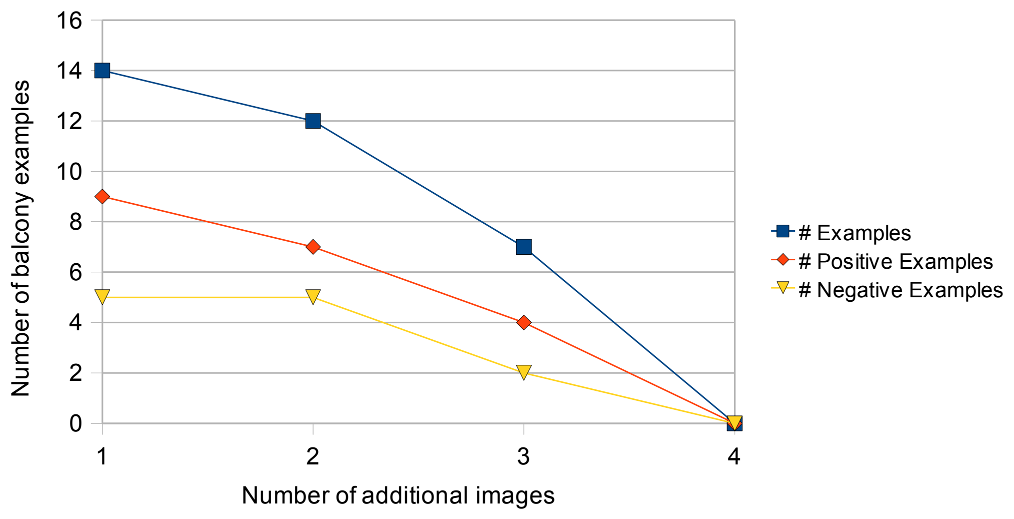
The complete system and experimental setup is described in [Hartz et al., 2009]. The evaluation loop is briefly summarised here:

1. initialise all the parts of the system,

2. start the interpretation by supplying a partial interpretation to the high-level system, which is based on the annotation, but which excludes all instances of a particular class C,

3. perform the high-level interpretation, during which all objects other than those from class C are confirmed by the Matchbox,

4. compare the resulting interpretation with the intended interpretation from the database by passing them to an external evaluator,



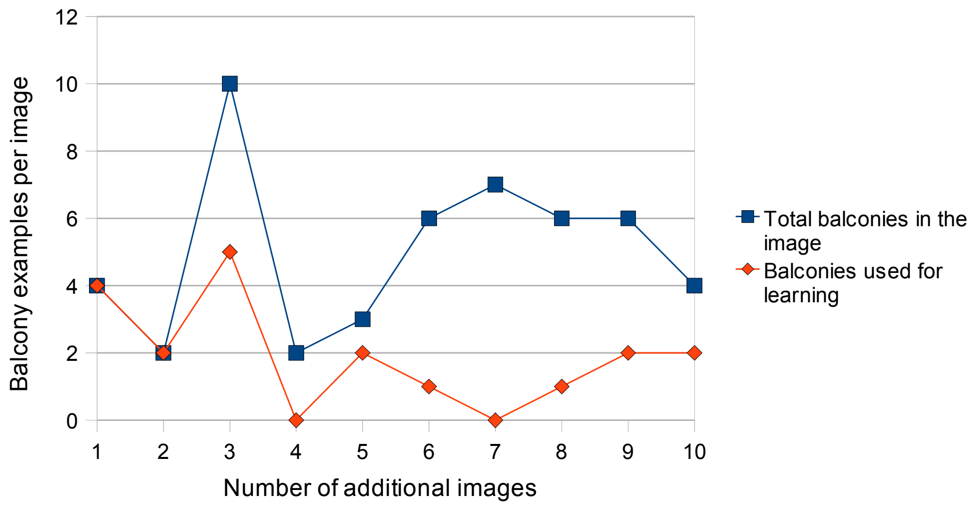


FIGURE 5.3: Top: Reducing needed positive and negative examples when proceeding with feedback learning. Bottom: Number of examples per image for learning a "Balcony" concept. The upper line depicts the number of balconies in each image, the lower line the number of used balcony examples for learning.

5. update the learned model to include the instances of class C which were not detected.

The result is an incremental learning loop which starts with a single image and improves the model with new images, thus learning from experience. Some results are shown in Figure 5.3.

### 5.2.2 Interpreting Structured Data

Man-made scene feature highly structured data, both in terms of visual structure and the compositional structure. An experiment was performed using a structural component detection algorithm for window detection as an image processing module within the

SCENIC system. It detects *structural components* by starting with seed detections provided by a window detector, and then imposing alignment rules by using an attributed grammar and comparing the ROI at expected window positions with the seed. The problem with this approach is that it leads to a number of conflicting and overlapping structural components, and many false positives. Each of the the detected structural components explains *some* of the windows, but usually none of them explains *all* the windows.

A crisp high-level interpretation system can be used to merge these detections into a coherent interpretation of the scene based on a domain-specific knowledge base. However, crisp interpretation depends on correct detections. In this case, the Matchbox was used to combine the two sources of structure. Detailed description of the the approach can be found in [Terzić et al., 2010]. Here, the experiment is summarised with the focus on the middle-level component.

The integration approach used was to integrate the structured evidence by starting with strong evidence and relying on top-down inference to fill in the missing objects. The algorithm is sketched below:

1. select the strongest evidence from the structured bag of evidence by the Matchbox,
2. interpret the image based on the available evidence,
3. suggest hypotheses of missing objects,
4. match the hypotheses to existing unused evidence.

Steps 3-4 are repeated until all hypotheses are checked and no more hypotheses can be made. The structural information from the low-level. which arranges objects into structures, helps the high-level to instantiate the correct aggregates and thus propagate the necessary constraints. The use of high-level hypotheses means that high-level context is used to pick out the correct evidence from a set of conflicting and spurious detections. This way, both sources of context are combined to create a single interpretation

The low-level evidence consists of many structural components, which in turn consist of many individual window detections. The window detections often overlap with detections from other structural components. We define the confidence $C_w$ of a window detection $W_n$ as

$$C_w(W_n) = \sum_{n \neq m} \frac{Area(W_n \cap W_m)}{max(Area(W_n), Area(W_m))}$$

Based on the confidences of individual window detections $W$, the confidence $C_r$ is calculated for each window row $R$ as:

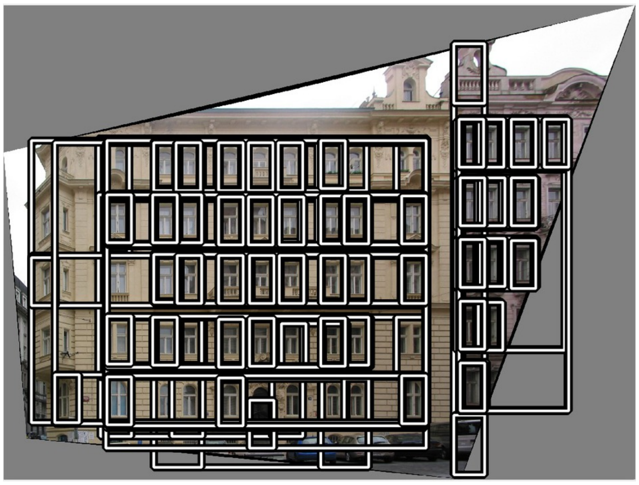


FIGURE 5.4: All the evidence from the low level. Many of the windows are detected several times.

$$C_r(R) = \sum_{W_n \in R} C_w(W_n)$$

If all $C_r$ are normalised to between 0 and 1, they present a heuristic approximation of the probability that a row detection matches to the "horizontal-alignment" high-level concept. The rows are then sorted according to their confidence, and a best set of non-overlapping rows is selected as initial evidence for interpretation by using a greedy algorithm. The row with the highest confidence $C_r$ is selected as the best row and all rows overlapping with that row in the vertical dimension are removed. Then the row with the second highest confidence is selected, and the process continues until there are no rows left. Figure 5.5 shows the result of this algorithm on one example image.

The selected rows and the corresponding window detections are passed to the high-level system. It is important to note that the partonomical relations are also passed as evidence along with the instantiated views, i.e. each window belongs to a row formation along with other windows. This means that the high-level doesn't have to examine all possible combinations of low-level detections in order to find a structure. After the initialization, the high-level interpretation is started.

The hypotheses created as a part of the interpretation process need to be confirmed by evidence from the low level. The task of the Matchbox is to look for the unused evidence which can confirm the generated hypotheses. The hypotheses are described in terms of allowed ranges for the position and size. The position and the size ranges are computed at the high-level such that the created hypotheses satisfy all the constraints imposed by the high-level window-array model (e.g. vertically aligned windows of similar size, which do not intersect). The hypothesis is confirmed if there is evidence detected in the provided area which has not been passed to the high-level system during the initial step

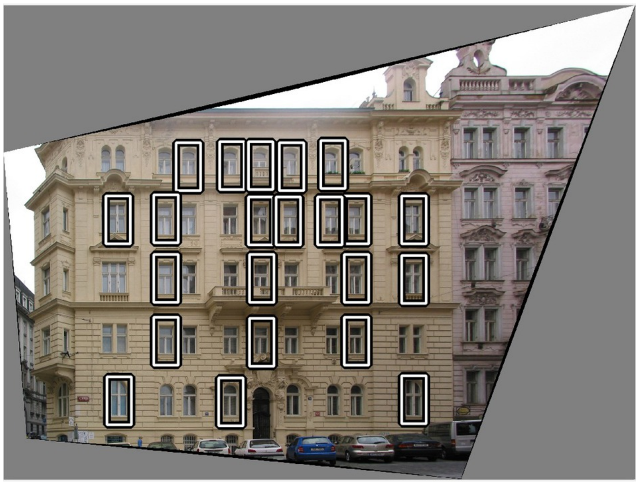


FIGURE 5.5: Automatically selected initial evidence. All horizontally aligned windows belong to the same rows (not shown here).

and which has a size allowed by the size ranges describing the hypothesis. If there are several possible matches, the match with the highest confidence $C_w$ is selected.

Due to the imperfection of image processing modules, the exact position and size of the detections is extended to ranges of allowed values. These uncertainty ranges cover the observed inaccuracy of the IPM and were determined experimentally on a set of annotated images. The matching process then looks at the intersection of the position and size ranges of the hypothesis and the available evidence and confirms the hypothesis if the intersection of all ranges is not empty.

At the end of the interpretation, the unconfirmed hypotheses are discarded as hallucinations, and the combination of the evidence selected in the initial step and the confirmed high-level instances forms the final interpretation of the image.

The combined system was tested on 7 hand-annotated images from the façade domain, consisting of 261 windows. Table 5.1 shows the effect of the combined structure models on the detection rate. The selected low-level detections used for initialisation (third column) are tested against the annotation. The detection rate of this bottom-up approach is shown in the sixth column as a baseline for comparison. If the confirmed high-level hypotheses (fifth column) are added, the detection rate generally improves (seventh column).

The different steps of the process can be seen in Figure 5.6. An interesting observation is that a number of hypotheses which correspond to windows in the image, shown in (d) are not among the hypotheses which were confirmed by the low-level (e). The main cause for this was poor contrast and occlusions which prevented some of the windows from being detected by the low-level stage. In other words, there were no window detections in any of the structural components which corresponded to these windows.

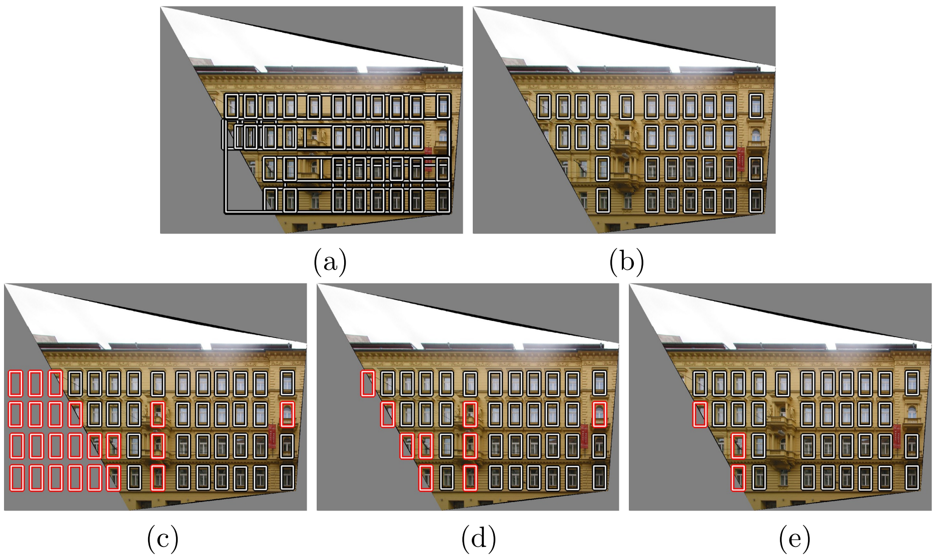
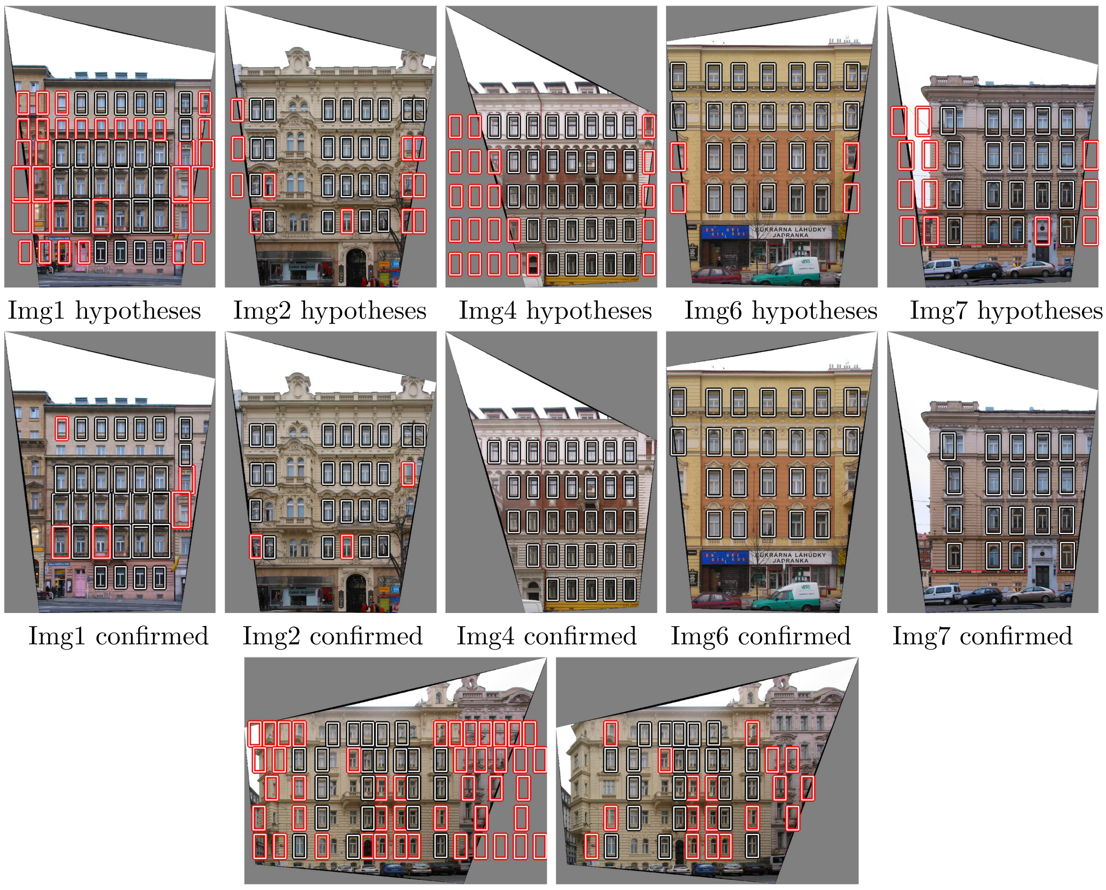


FIGURE 5.6: The interpretation process shown on image 3. (a) all evidence provided by the structure detector, (b) the evidence used to initialise the high-level system, (c) all hypotheses (shown in red), (d) the hypotheses which correspond to real objects and (e) the hypotheses which were automatically confirmed by low-level evidence. The windows partially occluded by balconies were hypothesised correctly, but there were no low-level detections to confirm them.

TABLE 5.1: Improvement of the window detection rate after combining two structure sources. In images 4, 6 and 7, the low-level structure detector detected everything except partial windows at the edge of the image, so there was no improvement.

| Img | ann. win. | correct win. det. | correct hypo. | confirmed corr. hypo. | low-level det. rate | combined det. rate | improvement | combined false pos. |
|---|---|---|---|---|---|---|---|---|
| 1 | 44 | 27 | 13 | 5 | 61.4% | 72.7% | 11.3% | 0 |
| 2 | 37 | 23 | 7 | 3 | 62.2% | 70.3% | 8.1% | 0 |
| 3 | 40 | 33 | 9 | 3 | 82.5% | 90% | 7.5% | 0 |
| 4 | 35 | 33 | 4 | 0 | 94.3% | 94.3% | 0% | 0 |
| 5 | 60 | 22 | 15 | 15 | 36.7% | 61.7% | 25% | 3 |
| 6 | 25 | 24 | 4 | 0 | 96% | 96% | 0% | 0 |
| 7 | 20 | 19 | 0 | 0 | 95% | 95% | 0% | 0 |


Img1 hypotheses Img2 hypotheses Img4 hypotheses Img6 hypotheses Img7 hypotheses

Img1 confirmed Img2 confirmed Img4 confirmed Img6 confirmed Img7 confirmed

Img5 hypotheses Img5 confirmed

FIGURE 5.7: The result of the combined structured models on the testing images.

Figure 5.7 shows the results on the remaining images. In three cases, there was no improvement, since the initially selected evidence has already detected all the windows. The created hypotheses were not confirmed. Image 5 (shown at the bottom) was particularly challenging due to the rich and irregular structure. The confirmed high-level hypotheses detected 15 further windows compared to the bottom-up approach, but also found 3 false positives. The evaluation data show that the combined approach improved the detection considerably on some images (especially images 2, 3 and 5). At the same time, it didn't hurt the cases where the low-level approach was already successful (images 4, 6 and 7). Our combined approach has only resulted in false positive detections in one case (image 5).

False positives can only occur if both structure models expect an object at a wrong location. Since both structure models rely on regularity assumptions, this sometimes occurs on images with an irregular structure. However, as can be seen in Figure 5.6 (c), the combination of the two structure sources makes good hypotheses that are confirmable by a human. Therefore, there is potential for further improvement of the detection rate if additional image processing modules are integrated into the system and used for confirming hypotheses.

TABLE 5.2: The means of the Gaussian distributions used to generate synthetic samples.

| | | | |
|---|---|---|---|
| $\mu(class1)$ | -0.5 | -0.5 | -0.5 |
| $\mu(class2)$ | 0.5 | 0.5 | 0.5 |
| $\mu(class3)$ | -0.5 | -0.5 | 0.5 |
| $\mu(class4)$ | 0.5 | 0.5 | -0.5 |
| $\mu(class5)$ | -0.5 | 0.5 | -0.5 |
| $\mu(class6)$ | -0.5 | 0.5 | 0.5 |
| $\mu(class7)$ | 0.5 | -0.5 | 0.5 |
| $\mu(class8)$ | 0.5 | -0.5 | -0.5 |

## 5.3 Evaluation of Probability Estimates

One way for estimating class probabilities presented in this thesis was by using automatically learned decision trees. There are other ways to estimate class probabilities which were implemented as a part of the Matchbox, but the decision trees have the advantage that they can describe regions of feature space using a compact representation, because they subdivide the feature space into regions with axis-parallel boundaries. This property makes it easy to pass down expectations to the low-level, as described in Section 3.6.1.3. The performance of the middle layer depends on accurate probability estimates. This section evaluates the probability estimates provided by decision trees.

### 5.3.1 Classification on Synthetic Data

The decision trees were first tested on synthetic 4-class and 8-class data. Each sample is drawn from a 3-dimensional Gaussian distribution. Table 5.2 shows the means of the distributions, and Table 5.3 shows the standard deviations and the priors of the classes. The first dataset has four equally probable classes, the second set has class priors chosen to be more similar to the façade domain, and the third set increases the standard deviation leading to more overlap between classes. Datasets 4 to 6 follow the same pattern, using 8 classes.

The results were compared with SVM-based multiclass classifiers using the svmlight software [Joachims, 1999]. For each $N$-class dataset, we obtained three results: using the learned decision tree followed by a MAP classification, by choosing the strongest response from $N$ one-against-all SVM classifiers (which we refer to as *SVM1*), and finally by performing a majority vote among $N(N-1)/2$ pairwise SVM classifiers (referred to as *SVM2*). The 4-class datasets were tested using 10000 training samples (of which 1000 are used for pruning the trees) and 10000 test samples. The 8-class datasets used 20000 training samples (2000 for pruning) and 20000 test samples.

TABLE 5.3: The priors on the classes of the synthetic data and the standard deviation used for the Gaussian distributions modelling the classes.

| | DS 1 | DS 2 | DS 3 | DS 4 | DS 5 | DS 6 |
|---|---|---|---|---|---|---|
| P(class1) | 0.25 | 0.1 | 0.1 | 0.125 | 0.05 | 0.05 |
| P(class2) | 0.25 | 0.1 | 0.1 | 0.125 | 0.05 | 0.05 |
| P(class3) | 0.25 | 0.2 | 0.2 | 0.125 | 0.05 | 0.05 |
| P(class4) | 0.25 | 0.6 | 0.6 | 0.125 | 0.05 | 0.05 |
| P(class5) | 0 | 0 | 0 | 0.125 | 0.05 | 0.05 |
| P(class6) | 0 | 0 | 0 | 0.125 | 0.10 | 0.10 |
| P(class7) | 0 | 0 | 0 | 0.125 | 0.10 | 0.10 |
| P(class8) | 0 | 0 | 0 | 0.125 | 0.55 | 0.55 |
| $\sigma_{1-8}$ | 0.5 | 0.5 | 1.5 | 0.5 | 0.5 | 1.5 |

TABLE 5.4: Comparison of a one-against-all SVM classifier (SVM1), a pairwise SVM classifier (SVM2), and our decision tree on 6 synthetic datasets. The numbers represent the classification rate for a given dataset.

| | SVM1 | SVM2 | Decision tree | |
|---|---|---|---|---|
| DS1 | 0.7805 | 0.7801 | 0.7654 | (73 nodes) |
| DS2 | 0.8395 | 0.8412 | 0.8311 | (109 nodes) |
| DS3 | 0.6843 | 0.6863 | 0.6745 | (119 nodes) |
| DS4 | 0.5825 | 0.5915 | 0.5829 | (411 nodes) |
| DS5 | 0.7238 | 0.7236 | 0.7145 | (335 nodes) |
| DS6 | 0.5604 | 0.5793 | 0.5722 | (25 nodes) |

The results, shown in Table 5.4 show the comparison of our approach with the SVM-based classifiers. The performance is within a percentage point of the SVM classifiers in almost all cases, outperforming one of the SVM classifiers on datasets 4 and 6.

### 5.3.2 Classification on Building Data

Our experiments on real data are based on the annotated façade image database from the eTRIMS project. All images are fully annotated using bounding polygons and class labels from a common ontology. In this particular experiment, an extended set of 599 annotated images was used, consisting of 27922 objects in total. From these images, 15357 were used as training objects, 6981 as validation objects for pruning, and 5584 as testing objects.

Table 5.5 shows the composition of the 18-dimensional feature vector used to describe each object. Simple and general features were used, because previous work on feature selection showed these features to be useful in the façade domain [Drauschke and Förstner, 2008a,b], and more complex features such as statistical moments and colour histograms did not perform as well in previous experiments.

TABLE 5.5: The composition of the feature vector.

| | |
|---|---|
| $f_0$ | area |
| $f_1$ | compactness: $4\pi \times area/perimeter^2$ |
| $f_2$ | aspect ratio: $width/height$ |
| $f_3$ | rectangularity: $area/(width \times height)$ |
| $f_{4-5}$ | mean and standard deviation of the red channel |
| $f_{6-7}$ | mean and standard deviation of the blue channel |
| $f_{8-9}$ | mean and standard deviation of the green channel |
| $f_{10-18}$ | 8-bin edge orientation histogram |

The results of the decision tree classifier learned for the 24-class façade object problem using the Gini coefficient for optimisation can be seen in Figure 5.8. The overall classification rate across all classes is 75.63%, with most classes showing a strong peak at the diagonal of the confusion matrix (see Figure 5.8).

It is apparent that the classes *Facade* and *Building* are often confused, as are *Road* and *Pavement*, but this is an expected result, given how visually similar these classes often are. This is a point where high-level context (in terms of a prior expectation for the classes) could improve the classification results.

Another interesting result is the poor performance with classes *Sign, Chimney* and *Door*. In the case of *Sign* and *Chimney*, the prior of the classes is so low that classifying all of them as windows actually reduces the overall error rate. The prior of the class *Door* is quite high but the visual appearance of doors is often very close to the appearance of the *Window* class, which has a far higher prior. The solution to these problems is to introduce contextual information in the form of updated priors for different image regions. If there is a strong scene context suggesting one class over the other, this can be used for disambiguation, as will be shown in Section 3.2.2.

Once again, the results were compared with SVM-based multiclass classifiers using the svmlight software. Two SVM-based classifiers were used: based on 24 one-against-all SVM classifiers (SVM1), and based on 276 pairwise SVM classifiers (SVM2). All three tests were performed on exactly the same objects, using the same features to keep results comparable. The only difference was that all individual features were scaled to between 0 and 1 for the SVMs. Since SVMs do not need a validation set, the objects used for pruning the decision tree were used as additional training objects for the SVMs. The default kernel (radial basis function) and default parameters (determined automatically by the svmlight software) were used.

Table 5.6 shows the results. The decision-tree based method outperforms both SVM-based methods in bottom-up classification. The confusion matrices for the SVM-bases classifiers are shown in Figures 5.9 and 5.10. Another interesting observation is that the problems with classification of doors are even more pronounced when using SVM-based

TABLE 5.6: Comparison of a one-against-all SVM classifier (SVM1), a pairwise SVM classifier (SVM2), and the decision tree on 5584 objects from 599 annotated images from the façade domain.

| SVM1 | SVM2 | Decision tree | |
|---|---|---|---|
| 0.7092 | 0.6999 | 0.7563 | (601 nodes) |

classifiers, as opposed to decision trees. The performance on the *Balcony* class is also worse.

To be fair, the default parameters used for the SVM classifiers might not be optimal. If a proper grid search through parameter space had been performed using a multi-class SVM implementation such as [Chang and Lin, 2001], the performance would likely be at least as good as that of the decision tree. Such a grid search can be done automatically, but it is costly in terms of processing time. What is important to note, though, is that the decision tree learned in this way has no parameters to tune, and outperforms SVM implementations when using default parameters, which makes them an attractive way to estimate probabilities when the feature vectors are low-dimensional.

### 5.3.3 Accuracy of Probability Estimates

One nice property of decision trees is that they provide an estimate of the probability of correct classification (without consideration of context). In scene interpretation systems, this is useful information since it can be used to influence the order of interpretation steps. However, it is well-known that when trees are learned in a way that tries to maximise the classification rate, the probability estimates are incorrect, especially for domains with unbalanced priors [Zadrozny and Elkan, 2001].

Several *probability smoothing* approaches have been proposed in the literature to address this problem [Bahl et al., 1989, Buntine, 1992, Zadrozny and Elkan, 2001]. A common and effective smoothing approach is $m$-estimation introduced by Cestnik [Cestnik, 1990]. The probability estimate at the leaves $P(c|l) = \frac{N_c(l)}{N(l)}$ is replaced by $P_s(c|l) = \frac{N_c(l)+P_d(c)m}{N(l)+m}$, where $P_d(c)$ is the domain prior for class $c$. The smoothing was applied to the tested decision trees by calculating the smoothed probabilities $P_s(C|L)$ for all classes and leaves. Since m-smoothing is a heuristic which affects different classes differently, the probabilities were renormalised in all leaves in the end so they sum to one again. The parameter $m$ determines how strongly the probabilities at the leaves are adjusted towards the domain prior. The parameter $m$ was determined experimentally, as described below.

The estimated probability provided by the leaves of the learned decision trees was compared with the actual classification rate. To this end, a tree with no smoothing was compared to a number of trees corresponding to different values for the parameter $m$.

| | Balcony | Building | Canopy | Car | Chimney | Cornice | Door | Dormer | Entrance | Facade | Gate | Ground | Pavement | Person | Railing | Road | Roof | Sign | Sky | Stairs | Vegetation | Wall | Window | Window-Array |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Balcony | 106 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 5 | 2 | 0 | 0 | 1 | 0 | 5 | 0 | 16 | 0 | 0 | 0 | 7 | 0 | 77 | 9 |
| Building | 9 | 79 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 72 | 0 | 0 | 0 | 0 | 1 | 0 | 11 | 0 | 1 | 0 | 6 | 0 | 9 | 5 |
| Canopy | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | 2 | 1 |
| Car | 1 | 0 | 0 | 53 | 0 | 0 | 2 | 4 | 4 | 1 | 0 | 0 | 1 | 0 | 7 | 2 | 10 | 0 | 0 | 0 | 13 | 0 | 23 | 0 |
| Chimney | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 13 | 0 |
| Cornice | 0 | 0 | 0 | 0 | 0 | 192 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 |
| Door | 0 | 0 | 0 | 0 | 0 | 0 | 49 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 149 | 2 |
| Dormer | 5 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 5 | 1 |
| Entrance | 3 | 0 | 0 | 1 | 0 | 0 | 4 | 0 | 14 | 4 | 0 | 0 | 0 | 0 | 2 | 0 | 3 | 0 | 0 | 0 | 2 | 0 | 40 | 4 |
| Facade | 8 | 43 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 145 | 0 | 0 | 0 | 0 | 1 | 0 | 5 | 0 | 0 | 0 | 3 | 0 | 10 | 6 |
| Gate | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Ground | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Pavement | 4 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 15 | 0 | 9 | 11 | 10 | 0 | 0 | 1 | 8 | 0 | 11 | 2 |
| Person | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Railing | 9 | 0 | 1 | 4 | 0 | 3 | 0 | 1 | 0 | 0 | 0 | 0 | 2 | 0 | 137 | 1 | 5 | 2 | 0 | 0 | 7 | 0 | 51 | 9 |
| Road | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 11 | 0 | 2 | 10 | 4 | 0 | 1 | 0 | 9 | 0 | 0 | 3 |
| Roof | 23 | 3 | 0 | 10 | 0 | 0 | 0 | 1 | 6 | 0 | 0 | 0 | 7 | 0 | 5 | 0 | 76 | 0 | 0 | 0 | 19 | 0 | 14 | 5 |
| Sign | 0 | 0 | 0 | 1 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 8 | 0 | 0 | 5 | 0 | 0 | 2 | 0 | 25 | 0 |
| Sky | 2 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 7 | 0 | 90 | 0 | 3 | 0 | 3 | 0 |
| Stairs | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 6 | 0 |
| Vegetation | 6 | 8 | 0 | 7 | 0 | 0 | 0 | 0 | 4 | 2 | 0 | 0 | 7 | 0 | 7 | 2 | 21 | 0 | 1 | 0 | 102 | 0 | 22 | 3 |
| Wall | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Window | 34 | 3 | 0 | 3 | 0 | 1 | 39 | 0 | 2 | 0 | 0 | 0 | 1 | 0 | 28 | 0 | 5 | 1 | 1 | 1 | 8 | 0 | 2976 | 17 |
| Window-Array | 4 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 14 | 0 | 0 | 1 | 0 | 21 | 0 | 1 | 1 | 0 | 0 | 2 | 0 | 54 | 172 |

FIGURE 5.8: Confusion matrix for the learned decision tree. Overall classification rate is 75.63%.

| | Balcony | Building | Canopy | Car | Cornice | Chimney | Door | Dormer | Entrance | Facade | Gate | Ground | Pavement | Person | Railing | Road | Roof | Sign | Sky | Stairs | Vegetation | Wall | Window | Window-Array |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Balcony | 28 | 2 | 0 | 4 | 0 | 1 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 2 | 0 | 2 | 0 | 1 | 0 | 2 | 0 | 5 | 0 |
| Building | 2 | 92 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 45 | 0 | 1 | 1 | 0 | 0 | 1 | 21 | 0 | 0 | 0 | 0 | 0 | 2 | 1 |
| Canopy | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Car | 1 | 0 | 0 | 47 | 0 | 0 | 0 | 0 | 1 | 2 | 0 | 0 | 4 | 0 | 0 | 3 | 6 | 4 | 0 | 1 | 1 | 0 | 0 | 0 |
| Cornice | 2 | 0 | 2 | 0 | 160 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 13 | 1 | 4 | 2 | 0 | 0 | 0 | 0 | 16 | 14 |
| Chimney | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Door | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Dormer | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Entrance | 4 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Facade | 3 | 49 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 120 | 0 | 0 | 1 | 0 | 1 | 3 | 3 | 0 | 0 | 0 | 0 | 7 | 0 | 16 |
| Gate | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Ground | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| Pavement | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 16 | 0 | 0 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Person | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Railing | 5 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 6 | 0 | 30 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 2 | 1 |
| Road | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 9 | 0 | 1 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Roof | 4 | 4 | 1 | 8 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 6 | 0 | 9 | 4 | 59 | 0 | 0 | 0 | 10 | 0 | 4 | 0 |
| Sign | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| Sky | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 105 | 0 | 0 | 0 | 3 | 0 |
| Stairs | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Vegetation | 9 | 23 | 0 | 10 | 0 | 0 | 5 | 0 | 12 | 16 | 0 | 1 | 8 | 0 | 6 | 10 | 34 | 0 | 1 | 2 | 151 | 0 | 14 | 1 |
| Wall | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Window | 166 | 17 | 4 | 50 | 35 | 12 | 200 | 16 | 57 | 25 | 1 | 0 | 19 | 0 | 165 | 3 | 36 | 38 | 4 | 13 | 26 | 0 | 3070 | 169 |
| Window-Array | 4 | 4 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 1 | 0 | 0 | 0 | 0 | 1 | 70 |

FIGURE 5.9: Confusion matrix for the one-against-all SVM classifier (SVM1). Overall classification rate is 70.92%.

| | Balcony | Building | Canopy | Car | Cornice | Chimney | Door | Dormer | Entrance | Facade | Gate | Ground | Pavement | Person | Railing | Road | Roof | Sign | Sky | Stairs | Vegetation | Wall | Window | Window-Array |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Balcony | 3 | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 4 | 0 | 209 | 3 |
| Building | 6 | 84 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 42 | 0 | 0 | 0 | 0 | 0 | 0 | 12 | 0 | 0 | 0 | 13 | 0 | 32 | 4 |
| Canopy | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 5 | 4 |
| Car | 0 | 1 | 0 | 47 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 12 | 0 | 54 | 0 |
| Cornice | 0 | 0 | 0 | 0 | 158 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 35 | 3 |
| Chimney | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14 | 0 |
| Door | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 202 | 0 |
| Dormer | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 0 |
| Entrance | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 8 | 0 | 64 | 0 |
| Facade | 2 | 42 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 121 | 0 | 0 | 1 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 5 | 0 | 36 | 8 |
| Gate | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Ground | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Pavement | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 16 | 0 | 0 | 9 | 8 | 0 | 0 | 0 | 5 | 0 | 35 | 0 |
| Person | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Railing | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 3 | 0 | 210 | 3 |
| Road | 0 | 3 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 13 | 0 | 0 | 6 | 1 | 0 | 0 | 0 | 6 | 0 | 9 | 1 |
| Roof | 5 | 14 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 5 | 0 | 0 | 0 | 62 | 0 | 0 | 0 | 22 | 0 | 50 | 5 |
| Sign | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 40 | 1 |
| Sky | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 103 | 0 | 1 | 0 | 4 | 0 |
| Stairs | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 14 | 0 |
| Vegetation | 2 | 7 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 16 | 0 | 0 | 0 | 122 | 0 | 37 | 0 |
| Wall | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| Window | 0 | 3 | 0 | 2 | 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 3 | 4 | 0 | 0 | 0 | 10 | 0 | 3079 | 4 |
| Window-Array | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 2 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 154 | 107 |

FIGURE 5.10: Confusion matrix for the pairwise SVM classifier (SVM2). Overall classification rate is 69.99%.

Table 5.7: Comparison of estimated class probabilities for the strongest class with the actual classification rate. Leaves with similar probabilities for the strongest class were grouped together in bins. The left column shows the expected result (mean value of each bin), and the other columns show the actually measured classification rates for these leaves. The best probability estimates were observed with a smoothed tree and m=1. In one case, no leaves had a probability estimate in the given range, this is indicated as "n/a".

| Expected | No smoothing | m=0.1 | m=0.5 | **m=1** | m=5 | m=10 |
|---|---|---|---|---|---|---|
| 0.95 | 0.92 | 0.94 | 0.94 | **0.95** | 0.95 | 0.95 |
| 0.85 | 0.81 | 0.84 | 0.85 | **0.87** | 0.89 | n/a |
| 0.75 | 0.65 | 0.67 | 0.74 | **0.75** | 0.82 | 0.88 |
| 0.65 | 0.49 | 0.50 | 0.62 | **0.64** | 0.72 | 0.71 |
| 0.55 | 0.45 | 0.44 | 0.50 | **0.60** | 0.67 | 0.54 |
| 0.45 | 0.36 | 0.36 | 0.43 | **0.42** | 0.45 | 0.27 |

Table 5.8: Comparison of classification rate for the original tree (left column) and trees smoothed with different values of m (right).

| No smoothing | m=0.1 | m=0.5 | m=1 | m=5 | m=10 |
|---|---|---|---|---|---|
| 0.7563 | 0.754835 | 0.752507 | 0.750895 | 0.708453 | 0.682307 |

Ideally, the probability estimate of the decision tree will be the same as the probability observed in practice. In other words, if an object is classified as a window with $P(window|l) = 0.7$, we expect that such a classification will be correct in 70% of the cases. In order to test this, the nodes with similar $P(c_{strongest}|l)$ were grouped together and the actual classification rate measured for each group.

Table 5.7 summarises the results. It shows the original tree (no smoothing) and smoothed trees using $m = 0.1$, 0.5, 1, 5 and 10. It can be seen that smoothing improves the probability estimates, and that the best results were achieved with $m$=1. One downside of smoothing is that it usually reduces classification accuracy. The effect of different smoothing factors on the classification rate is shown in Table 5.8. It can be seen that smoothing with $m$=1 doesn't impact classification rate strongly, and still significantly improves the probability estimates, making it the best choice for this domain and feature vector.

## 5.4 Evaluation of Context-Based Classification

The previous section evaluated the performance of appearance-based probabilistic classifiers which estimate $P(C|E)$ and perform MAP classification. The Matchbox can

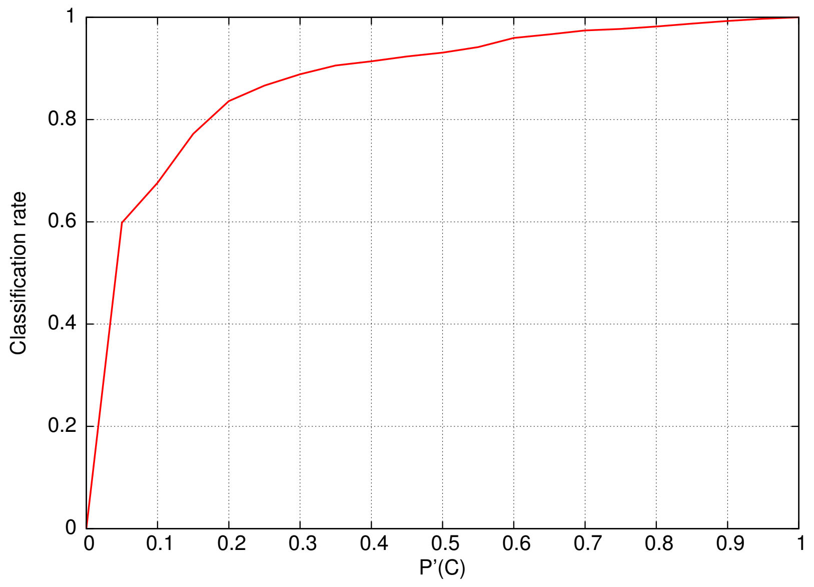


FIGURE 5.11: The effect of updated prior $P'(C)$ on the overall classification rate.

combine such appearance-based class probabilities with updated priors to improve performance. This section evaluates the classification improvement brought by introducing context in this way.

### 5.4.1 Simulated Context

First, the effect that changing the class priors has on the classification rate was tested. The correct scene context was simulated by artificially altering the priors $P(C)$. For each tested object, the prior on the correct class is set to a certain value $P'(C)$ and all other priors renormalised so they sum all up to one again.

#### 5.4.1.1 Effect of Context on Decision Tree Classification

Figure 5.11 shows the effect of updated $P'(C)$ on the overall classification rate. It can be seen that even small changes to the prior can have a great effect on the overall classification rate. As the prior for the correct class approaches one, the overall error tends towards zero, of course.

Figure 5.12 shows the effect on three different classes from the façade domain. The *Window* class has a very high domain prior (around 55%), the *Stairs* class has a very low domain prior (around 0.3%), and the *Door* class is relatively common (around 4%), but easily confused with the *Window* class. The graphs show that context is particularly helpful for less common and easily confused classes.

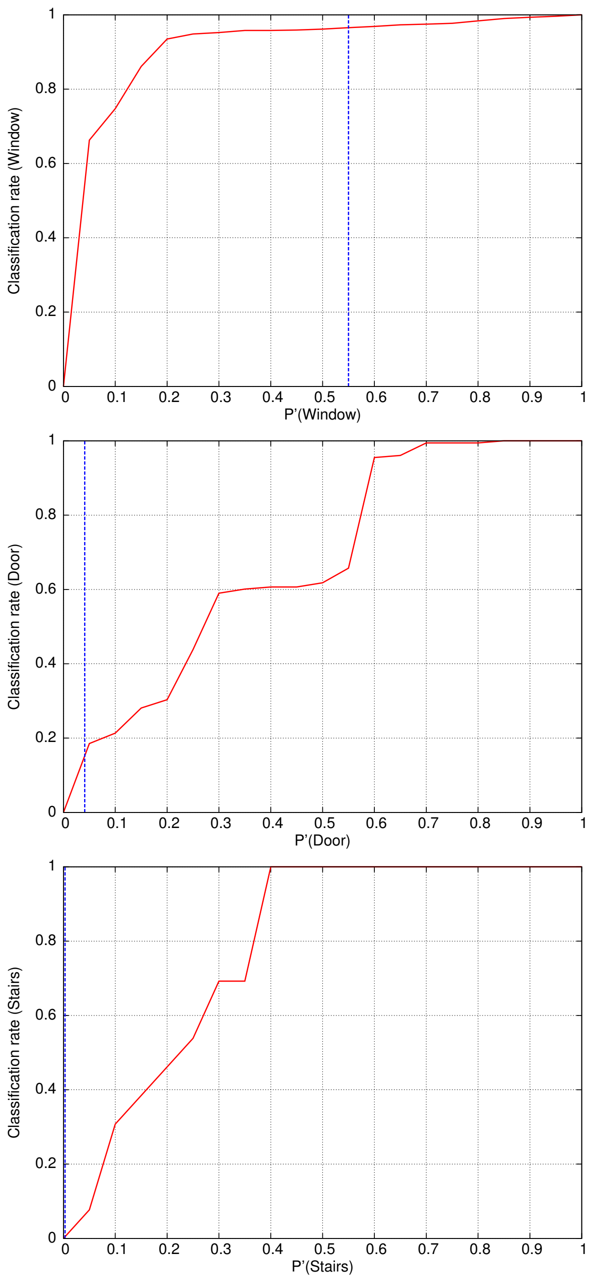


FIGURE 5.12: The effect of updated prior $P'(C)$ on three classes from the façade domain. From top to bottom, they are: *Window, Door* and *Stairs.* The vertical blue line shows the classification using the domain prior $P(C)$ without any change. The curves for door and stairs are jagged because they were obtained from fewer samples, which were represented by fewer nodes.

The context was simulated in these experiments, but it can be replaced by dynamic priors from Bayesian Compositional Hierarchies [Neumann, 2008] or a similar probabilistic reasoning scheme in the future. The improvements shown in Figures 5.11 and 5.12 suggest that scene context in the form of updated priors will lead to improved classification.

#### 5.4.1.2 Effect of Context on Structured Appearance Models

In order to test the effect of probabilistic context on a more complex, structured appearance model, a classifier based on [Ommer and Buhmann, 2010] (described in detail in Section 3.3.3) was implemented and used to provide class probabilities $P(C|E)$ for a selection of classes from the Caltech 101 dataset [Fergus and Perona, 2003]. The tested implementation uses SVMs for probability estimates (instead of NKDA) and uses the annotated bounding box information for determining the object centre, so only classification performance was tested. Only the context term $g^I$ and the first-order compositions $g_j, s_j$ were used in this experiment, the higher order compositions were not used.

The tested classifier is not as optimised and does not perform as well as the original algorithm, but this is not a problem since the goal of this experiment is to evaluate the effect of probabilistic priors on the classification rate. Figure 5.13 shows the baseline performance on the when using 30 images per class for training and the rest for testing. As in the original algorithm, the classes are assumed to be equally likely ($P(C) = 1/|C|$). The classification rate was 34.99%.

In the second experiment, Equation 3.18 was used to change the prior of each class to reflect the actual number of test images from each class. In other words,

$$P'(c) := N(c)/N_{all},$$

where $N(c)$ is the number of test images showing class $c$ and $N_{all}$ the number of all test images. This static prior reflects our knowledge about the distribution of the classes in this particular domain. Figure 5.14 shows the confusion matrix when the domain prior was used. The classification rate rises to 40.58%. It can also be seen that the classes “airplanes” and “watch” are heavily favoured, due to the very high prior probability for these classes. By favouring these two classes, the classifier reduces the total error, but these classes also have a higher rate of false positive classifications, as can be seen from the two vertical bars in the confusion matrix. Essentially, this experiment turned a Maximum Likelihood classifier (all classes assumed to be equally likely) into a Maximum a Posteriori classifier, without having to model the likelihood term $P(E|C)$ explicitly.

The second set of experiments with the structured appearance model replaced the static domain prior $P'(C) = 1/|C|$ by a dynamic prior boosting the correct class and thus simulating probabilistic scene context. As shown before with a decision tree classifier,

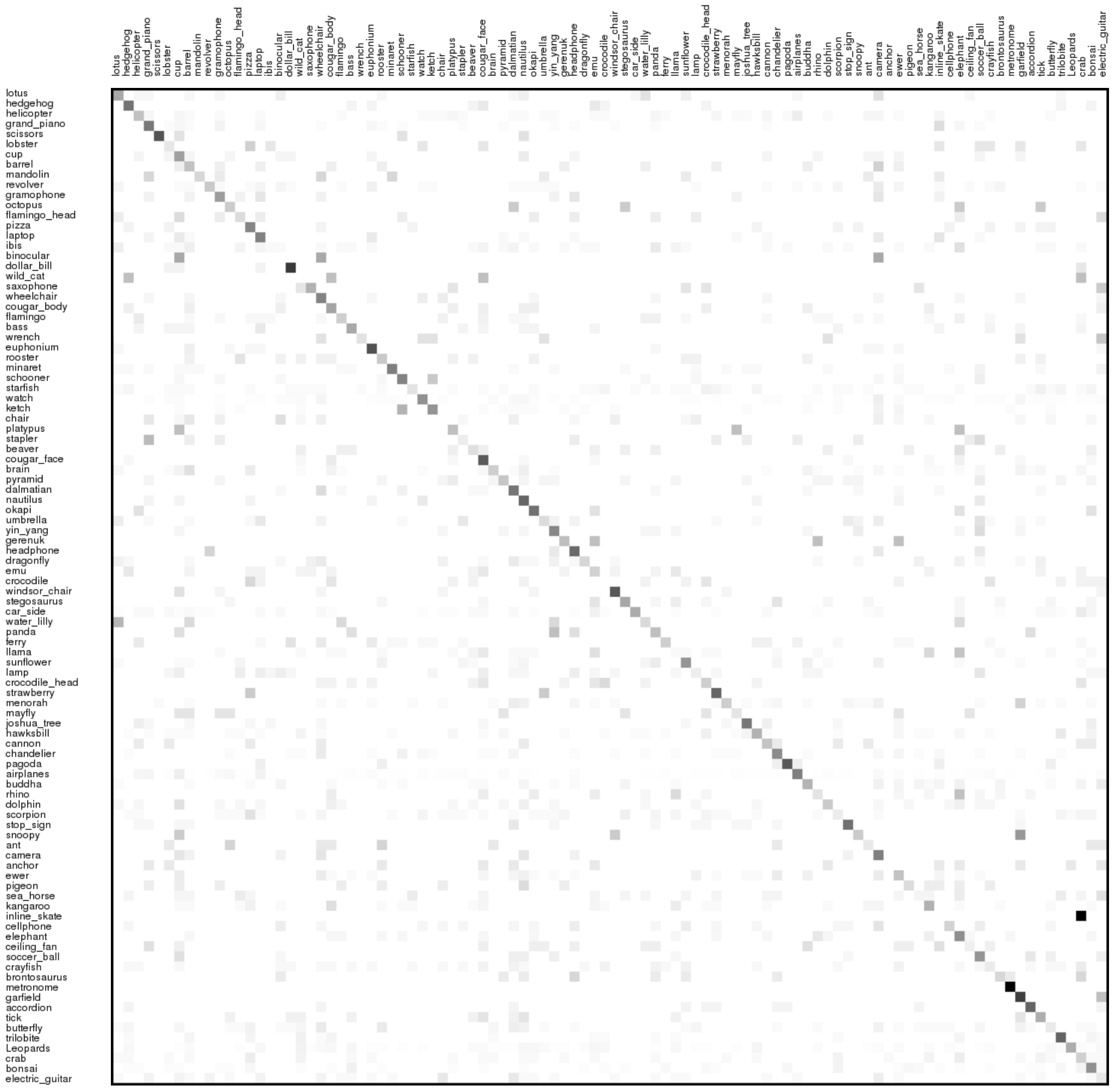


FIGURE 5.13: Confusion matrix (Maximum Likelihood). Class. rate = 34.99%. Although class names are difficult to read due to the size, the diagonal in the matrix is clearly visible.

the classification rate is evaluated for each class separately. The updated prior for the correct class $P'(c)$ varies between 0 and 1, and the other priors are scaled so all probabilities sum up to one. Figures 5.15 and 5.16 show the effect of the updated prior on the classes "cup" and "mandolin", respectively. It can be seen that even small increase of the prior of the correct class can significantly improve the classification rate.

### 5.4.2 Experiments Using Probabilistic High-Level Systems

The middle layer was further evaluated by using it as a part of a system for classifying regions from a database annotated facade images. The first experiment was performed on a sub-domain consisting of floors and used the BCH as a source of probabilistic context.

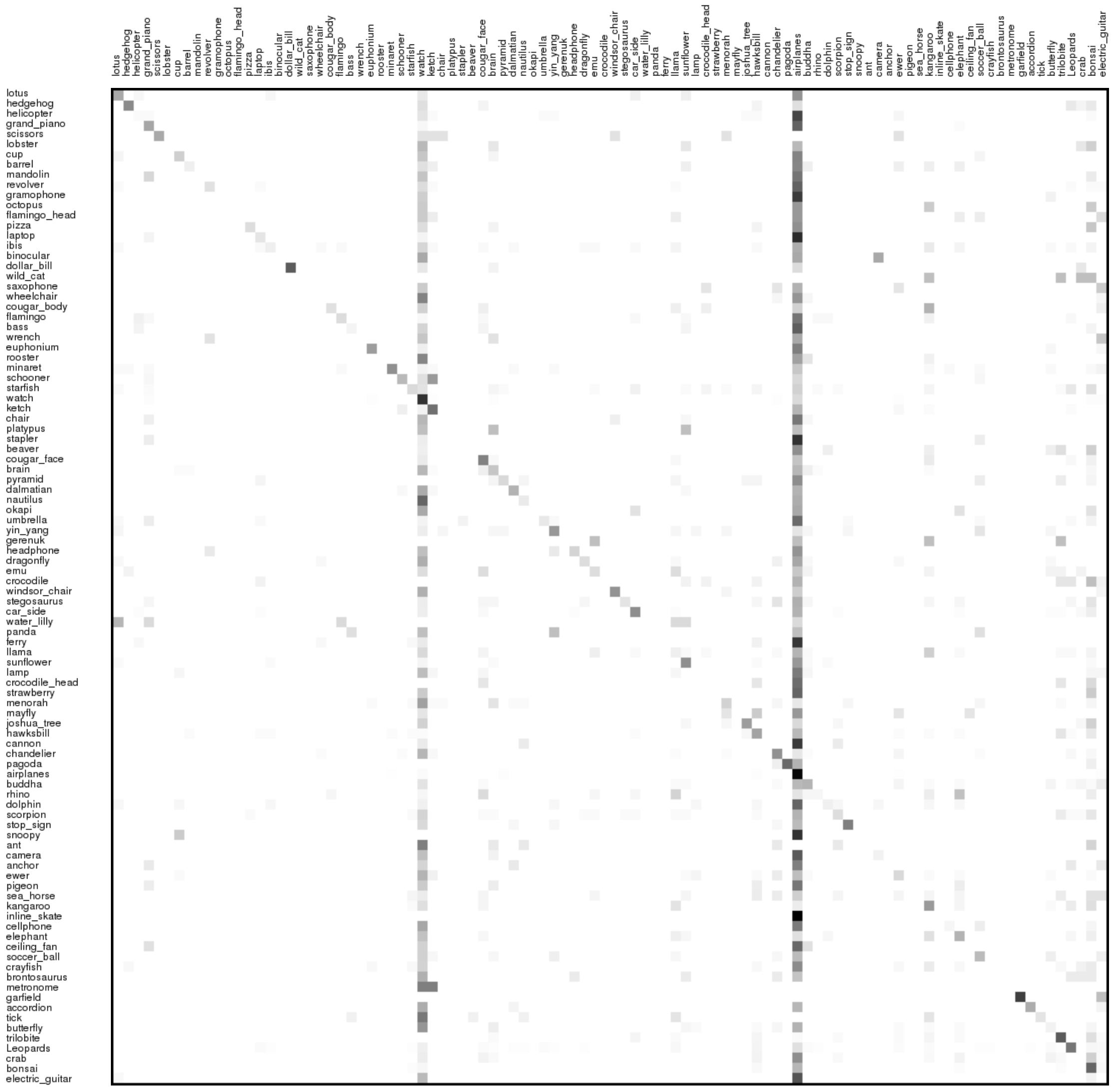


FIGURE 5.14: Confusion matrix (Maximum a Posteriori). Class. rate = 40.58%. The high prior for classes "watch" and "airplanes" results in a strong response for these classes, as evidenced by dark columns in the confusion matrix.

The second experiment was done on entire facades and compared the performance of the BCH against a neighbourhood-based Markov Random Field approach and a baseline using static domain priors.

### 5.4.2.1 Floor Domain

The domain consists of floors from the façade domain. 393 floors were automatically extracted from the eTRIMS annotated image database. Each object in the image is annotated with a bounding polygon and a class label. Figure 5.17 shows several examples from the floor database. The BCH was learned from the training set by drawing each sample several times, and applying uniform noice each time

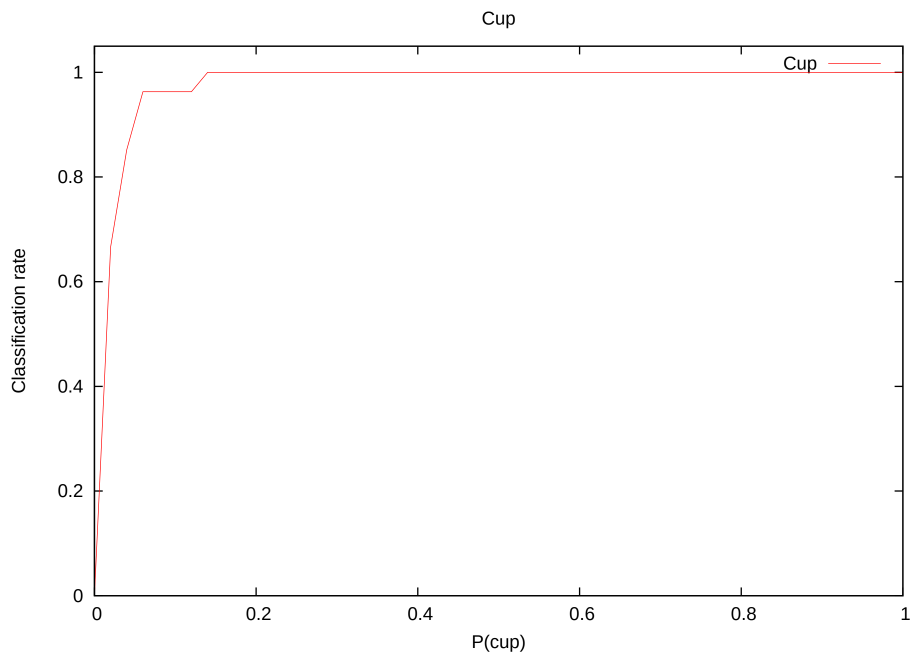


FIGURE 5.15: Prior adjustment for the cup class.

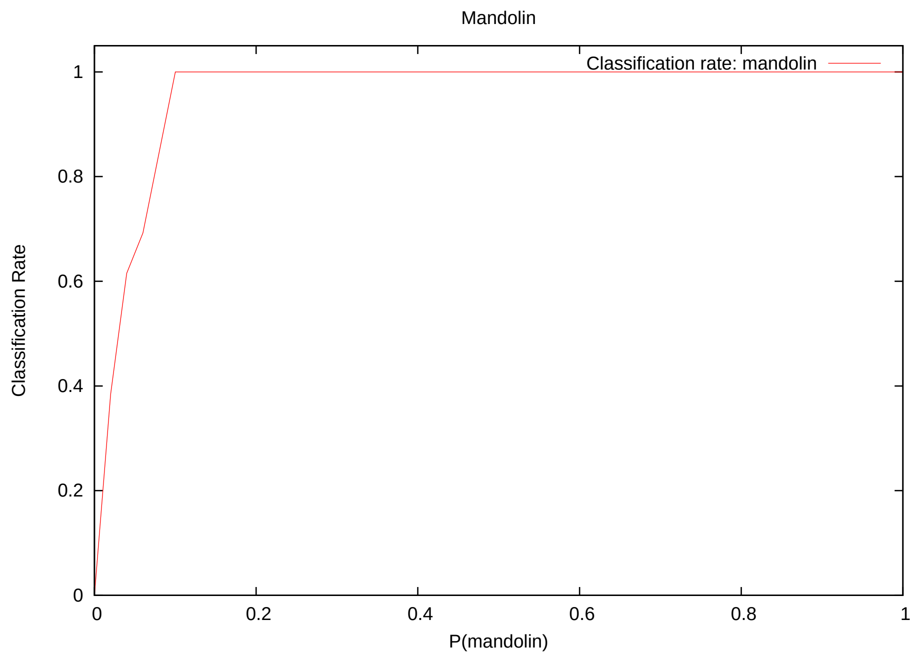


FIGURE 5.16: Prior adjustment for the mandolin class.

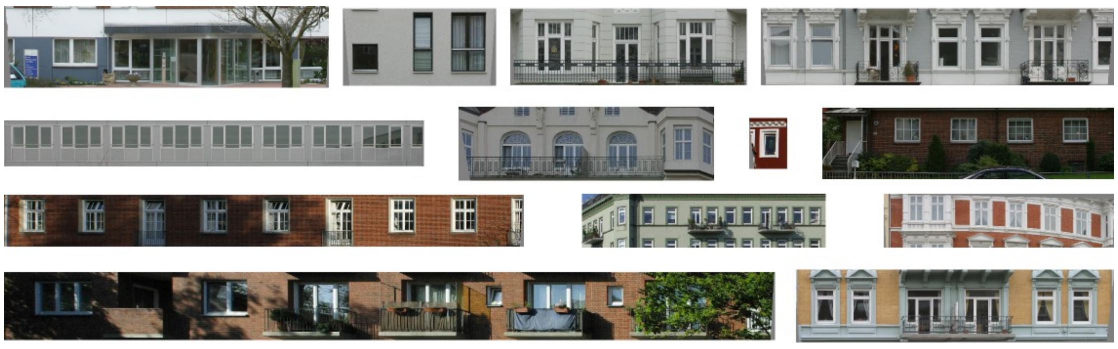


FIGURE 5.17: Examples of the variety of floors with the domain. Since the floors were automatically extracted using the annotations, they include floors which are only partially visible, such as the floor with only one window visible.

TABLE 5.9: A representative distribution of the classes within our data of the floors within an façade.

| **class** | **relative frequency** | |
|---|---|---|
| `Balcony` | 8.038 | % |
| `Canopy` | 0.629 | % |
| `Door` | 9.697 | % |
| `Entrance` | 1.287 | % |
| `Gate` | 0.114 | % |
| `Ground-Floor` | 1.831 | % |
| `Person` | 0.029 | % |
| `Railing` | 9.153 | % |
| `Sign` | 1.802 | % |
| `Stairs` | 0.486 | % |
| `Upper-Floor` | 9.411 | % |
| `Vegetation` | 0.057 | % |
| `Wall` | 0.057 | % |
| `Window` | 52.489 | % |
| `Window-Array` | 4.920 | % |

There were 15 object classes in the ontology:

- the primitives: `Door`, `Person`, `Sign`, `Wall`, `Window`, `Gate`, `Railing`, `Canopy`, `Stairs`, `Vegetation`
- functional entities consisting of several parts: `Entrance`, `Window-Array`, `Balcony`, `Ground-Floor`,`Upper-Floor`

Table 5.9 shows the relative frequencies of the classes in the extracted floors. The annotation includes a rough estimate of the image scale, which makes it possible to estimate the size of the objects.

TABLE 5.10: Classification rate determined by 10-fold cross-validation.

| **fold** | **BCH** | **DT** | **BCH & DT** |
|---|---|---|---|
| 0 | 0,65 | 0,70 | 0,75 |
| 1 | 0,66 | 0,65 | 0,78 |
| 2 | 0,62 | 0,64 | 0,77 |
| 3 | 0,69 | 0,72 | 0,78 |
| 4 | 0,68 | 0,68 | 0,75 |
| 5 | 0,62 | 0,65 | 0,76 |
| 6 | 0,67 | 0,71 | 0,83 |
| 7 | 0,63 | 0,7 | 0,83 |
| 8 | 0,66 | 0,68 | 0,76 |
| 9 | 0,63 | 0,71 | 0,79 |
| **Average** | 0,652 | 0,685 | **0,780** |
| **Std. Dev.** | 0,025 | 0,029 | **0,029** |

The input for this experiment was a set of polygons based on manual annotations, which need to be classified by a combination of local (visual) and global (contextual) features. The task was to classify each polygon by assigning it a label from our ontology. Using the polygons from the annotations as evidence eliminates errors due to *false positives* and allows clear statements about the helpfulness of context.

The local features $E_a$ used in the experiments were area, aspect ratio, compactness and rectangularity. They were chosen to be very simple here for two reasons. First of all, they are fast to compute and, as shown in [Terzić and Neumann, 2009a, Drauschke and Förstner, 2008b], the façade domain is a difficult classification problem even with much more complex features. Secondly, using weak features helps to illustrate the strength of probabilistic modeling and the usefulness of context. The global features $E_s$ were the horizontal and vertical position.

The performance of the combined system was compared to the performance of the Bayesian Compositional Hierarchies (BCH) alone (using $E_s$) and the decision tree (DT) alone (using $E_a$) to evaluate the improvement obtained by exploiting context. The evaluation was performed using ten-fold cross-validation with the results shown in Table 5.10. The average improvement of the classification rate observed over all testing data was 13.8%, from 0.685 of the decision tree (local visual appearance only) to 0.780 with the help of the contextual model. This shows that scene context can significantly improve classification.

One observed problem was that incorrect classifications introduced early on result in incorrectly established context, which often leads to further incorrect classifications. Although the classification algorithm tries to minimize this effect by starting with the most reliable evidence, incorrect initial classifications still occur, and the classification

TABLE 5.11: A confusion matrix reduced to the most likely classes

| | Balcony | Door | Ground-Floor | Railing | Stairs | Upper-Floor | Window | Window-Array |
|---|---|---|---|---|---|---|---|---|
| Balcony | 151 | 2 | 0 | 1 | 0 | 4 | 83 | 5 |
| Door | 2 | 139 | 0 | 0 | 0 | 0 | 157 | 1 |
| Ground-Floor | 0 | 0 | 2 | 0 | 0 | 50 | 7 | 0 |
| Railing | 3 | 1 | 0 | 181 | 0 | 0 | 94 | 2 |
| Stairs | 0 | 0 | 0 | 4 | 2 | 0 | 10 | 0 |
| Upper-Floor | 0 | 0 | 7 | 0 | 0 | 274 | 12 | 4 |
| Window | 9 | 33 | 0 | 6 | 0 | 1 | 1596 | 13 |
| Window-Array | 1 | 0 | 0 | 4 | 0 | 4 | 57 | 92 |

performance on such images is very poor. These few images drag the classification rate down.

An excerpt from the confusion matrix over all of the ten folds is shown in Table 5.11. It can be seen that the ground floor is often confused with the upper floor. Since the difference between the two is almost purely contextual (as opposed to visual), it might be better to delay the classification until other objects provide a strong enough context.

#### 5.4.2.2 Whole Facades

The second experiment evaluated the use of probabilistic context in the highly varied domain of entire building facade scenes. Two sources of context capable of calculating class priors for regions of the image were used as a source of probabilistic context in this experiment:

- a neighbourhood-based Markov Random Field classifier [Heesch and Petrou, 2009], and
- the Bayesian Compositional Hierarchy [Neumann, 2008].

The MRF was learned based on assymetrical relations between neighbouring regions and can be considered to be a source of *local context*. The BCH was learned on the entire facade structures, so it can be considered to be a source of *global context*. Finally, the static domain prior for all classes was used as a baseline comparison. This represents the best performance in the case where no context is available.

Since the MRF were designed to work with 12 classes, the annotations were reduced to the 12 classes shown in Table 5.12.

Table 5.12: The twelve classes used for the experiment with whole façades

| | | |
|---|---|---|
| `balcony` | `facade` | `sky` |
| `chimney` | `ground` | `stairs` |
| `door` | `others` | `vegetation` |
| `dormer` | `roof` | `window` |

The class priors were combined with a global appearance model which was estimated by a decision tree, as described earlier. This appearance model represents the distribution of different unary measurements across all classes. The classification rate was evaluated using 20-fold cross-validation performed on a set of 193 annotated facade images. In each of the individual runs, all of the components of the system (BCH, MRF and the decision trees) learned from the same training set. The classification rate was compared between the following combinations:

**DT** the *decision trees* combined with the domain prior for all classes

**MRF** *MRF-Classifiers based on Regions* without any unary measures

**BCH** *Bayesian Compositional Hierarchy* without any unary measures

**DT+MRF** the combination of MRF with unary measures via the DT

**DT+BCH** the combination of BCH with unary measures via the DT

The results are shown in Table 5.13 and summarized in Figure 5.18. The decision trees using the fixed domain prior showed the best overall classification rate of 71.4%. The Bayesian Compositional Hierarchies had a classification rate of 51.5% on their own, and 70.2% when combined with the decision trees. The Markov Random Field had a classification rate of 46.4% on their own, and 61.0% together with the decision trees. the combination of the BCH and the DT was only superior to the DT with the fixed domain prior on a single fold out of the twenty tested folds (fold 15 with 74.9% compared to 72.1%).

| **DT** | **MRF** | **BCH** | **DT+MRF** | **DT+BCH** |
|---|---|---|---|---|
| $0.714 \pm 0.041$ | $0.46 \pm 0.12$ | $0.515 \pm 0.042$ | $0.610 \pm 0.068$ | $0.702 \pm 0.041$ |

Table 5.13: The classification rates for all methods in the 20-fold cross-validation (mean and standard deviation reported).

The results indicate that the extremely varied domain of entire building facades is difficult to model at the high level. The majority of the facade structures only occur on one image of the dataset, making it difficult to create a generic structural model of the entire facade. This resulted in worse performance of the BCH when compared to the experiment done on the simpler floor domain. On the other hand, modelling small parts

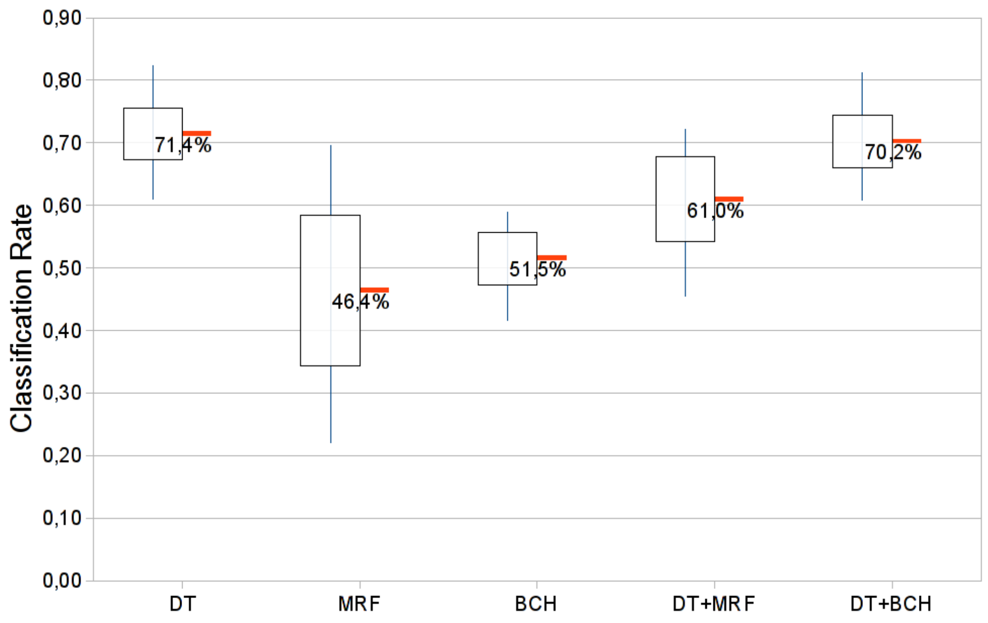


FIGURE 5.18: The average of each method is shown as well as the standard deviation (as boxes) and the minimum and maximum (as thin black lines).

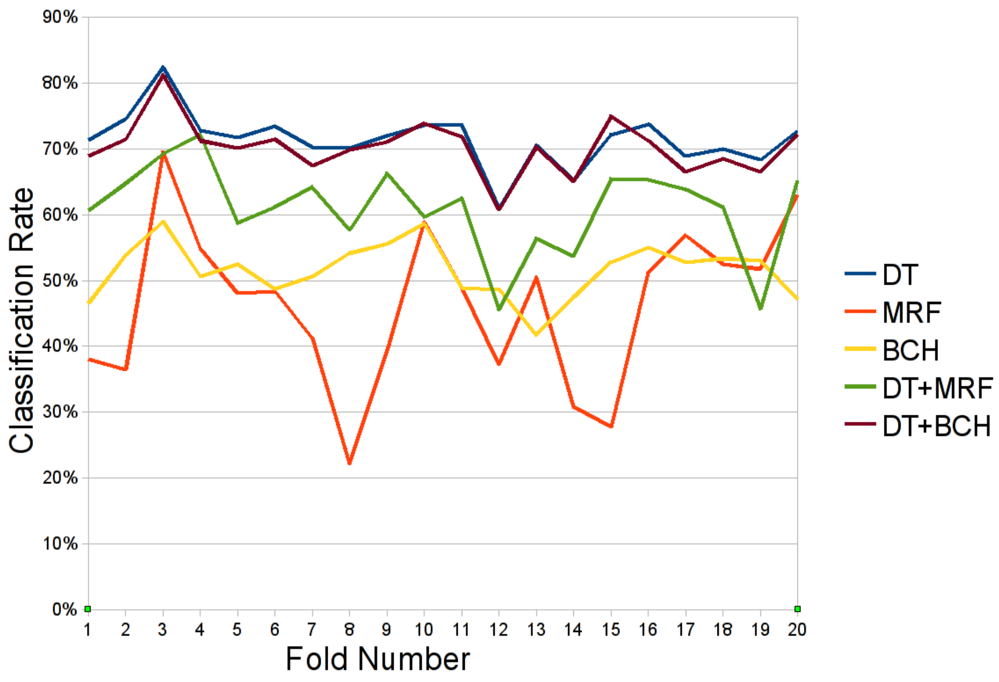


FIGURE 5.19: The classification rates for all methods in the 20-fold cross-validation

of facades in terms of neighbourhoods, as done with the MRF approach, does not seem powerful enough to provide strong probabilistic context, and the results in this case were worse than when using the static domain priors. One interesting observation relating to the BCH structures is that, given enough training examples, their performance tends to naturally approach that of static domain priors in difficult cases like these.

Despite the difficulties inherent in modelling difficult domains such as the facade domain, this experiment shows that the probability estimation performed by the middle layer can be easily used together with a wide range of probabilistic context sources, including a fall-back to static domain priors in the case where probabilistic context is not available. It can also be seen that this generic way of adding unary measurements to a contextual classifier always improves the overall performance.

### 5.4.3 Automatic Segmentation and Interpretation

The system was also evaluated as a source of context for segmenting a complete facade image. In this instance, the BCH contained six different probabilistic models of facade structures. Each of the six alternative models represented one particular structure. The goal of the experiment was to recognise the structure of a facade image and to use the evolving context to guide the segmentation process. Three separate segmentation algorithms were used for this experiment:

- a trainable window detector based on Adaboost [Terzić et al., 2010],[1]
- an interest region detector [Jahangiri and Petrou, 2008], and
- detector of stable regions in scale space [Drauschke, 2009].

Each one of the detectors was run on 190 images, and the segmentation results were automatically compared to the hand annotations by attempting to match each detected polygon to a manually annotated polygon. The polygon was considered to correspond to an annotated object if the overlap ratio:

$$\frac{A(\det \cap \text{ann})}{\max(A(\det), A(ann))}$$

was greater than 50%. In this case, the detected polygon was labelled with the correct class from the corresponding annotation. Otherwise, the polygon was labelled as a "False Positive" if there was no correspondence.

Then a feature vector was computed for each detected region. In adition to the four shape descriptors (area, rectangularity, compactness and aspect ratio), the means and standard deviations of the three colour channels (RGB) were used, giving a 10-dimensional

[1]The detector was written and provided by Jan Šochman, but the referenced paper contains the most detailed description available.

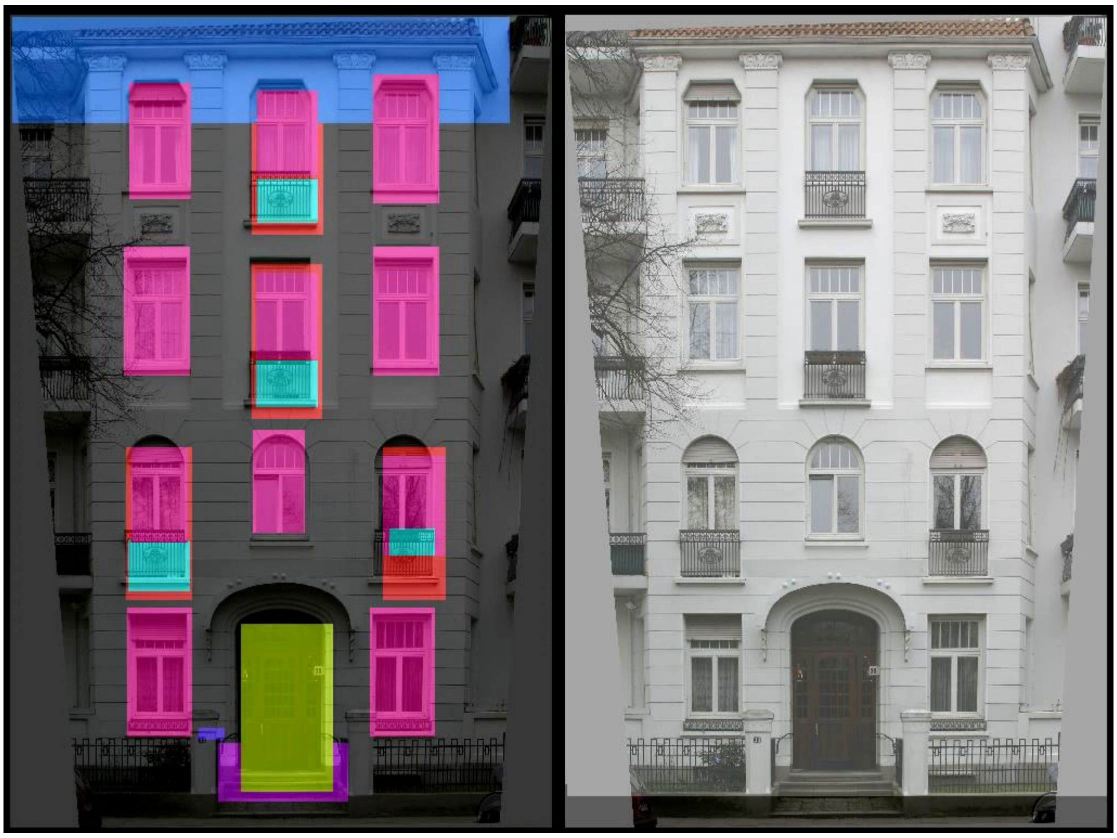


FIGURE 5.20: The result of context-based segmentation. The BCH used six structural models, out of which the most fitting one was automatically selected and used to interpret the soup of polygons provided by the low level.

feature vector describing each detected region. Then the class likelihood distrubution was modelled by a decision tree as described in section 3.3.2.1 for each image processing modules separately, including the likelihood that a false positive produced the detection.

The combination of the three IPMs managed to find many of the interesting objects in the scene (true positive detections), but more than 80% of detected regions were spurious (false positive detections). This experiment evaluated the improvement brought about by strong probabilistic context which can update class priors for different image regions based on the already integrated evidence. Like in the previous experiment, the region with the highest classification confidence was classified, and the context propagated, changing the class posteriors of the remaining regions.

Figure 5.20 shows the result of the final segmentation. Most regions were discarded as false positive detections, and the remaining detections were interpreted according to the most fitting of the six structural models. The experiment shows that strong context can be crucial in interpreting weak and conflicting low-level evidence. However, it remains to be seen how well the approach scales to hundreds of structural models.

## 5.5 Summary

The variety of experiments presented in this chapter show that the presented middle-layer architecture can be easily combined with a variety of low-level and high-level systems, allowing for many different combinations. It is also possible to combine multiple image processing modules to produce an improved segmentation.

Most experiments have shown that the combination of low-level and high-level processing using the proposed architecture improves performance. Experiments with really good low-level detections (hand-annotated regions without labels) showed that even very weak high-level context from the floor domain can significantly improve classification compared to a pure bottom-up approach (Section 5.4.2.1). Similarly, an experiment with a large number of weak detections (most of them spurious) showed that strong high-level context with few competing models can effectively aid segmentation (Section 5.4.3). It should, however, be noted that the cases where the high-level model is not strong enough because the domain is too varied (as in Section 5.4.2.2), or where the low level detections are too poor, still present a problem for the complete interpretation system.

# Chapter 6

# Conclusions

This thesis has presented a probabilistic middle-layer for image understanding. In contrast to some of the previous work, the presented approach does not introduce a new object-detection architecture. Instead, it presents a method for probabilistic context-passing for understanding static images, based on the insight from previous work on middle-layer architectures and modern object detection algorithms.

The middle layer is designed to function as a part of a larger image understanding system, so its performance can only be effectively measured when tested within such a complete system. However, there are inherent difficulties with measuring the performance of an interpretation system. The main problems are the lack of an objective measure which defines how "good" an interpretation is (especially when comparing different interpretation systems which use different internal representations), and the lack of large datasets specialising on image understanding (as opposed to object detection or classification).

Because of this, the performance was tested in two different ways. The evaluation of the middle layer in isolation was based on larger datasets, and was designed to test whether the central tasks of the middle layer are performed well: providing top-down detections with accurate probabilistic confidence values, and the effect of high-level context on object detection and classification performance. This is similar to the testing performed with other middle-layer architectures, such as those of Zhu and Mumford [2006] and Guhl and Shanahan [2007], who both demonstrated the feasibility of their approaches on object detection and segmentation, and not on complete scenes with complex relations between objects.

The second important aspect is the ability of the architecture to work as a part of a complete interpretation system. This was tested by using a number of high-level interpretation systems and low-level detectors, combined in different ways, and used to interpret complex scenes. The number of testing images was typically smaller in such experiments, but the results show that the Matchbox does function well as a part of a complete system for interpreting realistic images, and that it improves the overall interpretation.

## 6.1 Summary of Results

The problem statement introduced in Section 1.2 listed the requirements for a middle layer: it should be generic, provide an intermediate representation of the scene, model the uncertainty of the low-level algorithms, allow for using high-level scene context, support incremental interpretation, and be able to deal with occlusion and conflicts. Additionally, it should define the interface to low-level and high-level algorithms through a set of standardised services. This section discusses how well the proposed architecture fulfills these requirements.

**Generic design** The architecture allows for top-down and bottom-up interaction based on priors or object hypotheses with value ranges, which are common ways to provide feedback in high-level reasoning systems. The low-level detections are described in terms of class probabilities, which can be estimated on an annotated dataset. Because of this, the proposed architecture does not depend on a particular domain or algorithm. This was demonstrated by adapting a number of low-level and high-level systems to use the Matchbox as a bridge, which required relatively small adjustments.

**Intermediate-level representation** The scene is described in terms of object views (Section 3.2). A view is a structure representing the presence of an object in the image, which can be matched to high-level concepts used to reason about the scene. This object-centered representation corresponds to the concept of “percepts” in robotics, and follows the split between the scene domain and image domain used by many vision researchers.

**Uncertainty modelling** The low-level object detections are inherently uncertain. The proposed architecture models the uncertainty of object detection (false positives), object classification and the localisation and size errors using a probabilistic model learned on an annotated set for all low-level detectors (Section 3.3). This central model of detection confidence allows combining several detectors and communicates a detector-specific measure for the needed position and size tolerance to the high level.

**Integration of high-level context** The use of views provides the high-level reasoning systems with the ability to abstract from the details of the low-level algorithms. The integration of scene context is made possible by using a new context-based classification method which can combine appearance-based and structure-based class probabilities

(Section 3.2.2). Additionally, crisp hypotheses can be used to refine the classification by disallowing certain classes in the case where only static domain priors are available (Section 3.2.5).

**Incremental interpretation** The proposed architecture facilitates the incremental interpretation process in two ways. The probabilistic representation of classes makes it possible to start the incremental interpretation with a small number of very certain detections, reducing the danger of establishing wrong context early in the interpretation process. Also, the probabilistic context-based classification framework developed in Section 3.2.2 was designed with incremental interpretation in mind, and is capable of using evolving, dynamic prior probabilities for classifying the remaining objects.

**Middle-layer services** The middle layer services discussed in Chapter 3 and listed in Section 4.2.1 enable top-down and bottom-up flow of information. The middle layer can provide classified views on demand, or ask the high-level for priors to assist the classification of ambiguous views. This generic interface was used to interpret scenes using a number of low-level and high-level algorithms.

**Occlusion and conflicts** Dealing with occlusion and conflicts was discussed in Section 3.5.2. The proposed architecture uses the high-level knowledge to reason about allowed overlaps in the image plane, based on partonomical relations between classes and on allowed occlusion based on a special "can-overlap" relation. This type of modelling is admittedly simple, but is generic and can be applied to any domain. It allows for refuting hypotheses which conflict with observed evidence, confirming hypotheses which are supported by evidence, or delaying the matching in case the evidence neither supports nor conflicts with the hypothesis.

## 6.2 Open Issues

The middle-layer architecture is based around accurate probability estimates of object class, detection error, and the object's exact size and location. The experiments in this thesis have typically estimated these probabilities by using a large annotated dataset. Unfortunately, the existence of such a dataset can not always be taken for granted. While a number of shape and object category datasets exist, there are no large-scale scene interpretation benchmarks. Luckily, the localisation and classification probabilities can be obtained from the categorisation benchmarks. A good estimate of the detection error must be obtained on a more complex dataset. The VOC [Everingham, 2006] is such a dataset, but it is still simple in terms of structure compared to complex natural scenes.

In the case where good probability estimates are not available, they can be estimated using heuristics. The work on structured window detection presented in Section 5.2.2 used such a heuristic to measure the probability of window detections. This makes the architecture usable for many specialised domains where annotated datasets are not available, but expert knowledge about the scene is strong. It is clear, however, that this direction quickly undermines many of the advantages brought about by using calibrated probabilities.

In many image processing algorithms, optimal parameter selection is crucial. The automatically learned decision trees have no major parameters to tweak and have shown good performance when the number of low-level features is small. However, the estimates provided by SVMs (needed for hierarchical patch-based models) were very sensitive with respect to training parameters. Setting optimal parameters for SVM classifiers trained on a hundred classes is not an obvious task, and a poor choice of parameters can influence the performance of the entire system.

## 6.3 Comparison to Other Work

Compared to the blackboard middle-level architectures, such as the VISIONS system [Hanson and Riseman, 1978] and the architecture proposed by Guhl and Shanahan [2007], the representation proposed by the Matchbox is flatter, providing one additional layer of abstraction, at the level of object views. The blackboard architecture, which is based on autonomous agents, has a number of intermediate levels. It also highly parallel and allows for natural top-down and bottom-up information flow. On the other hand, the design based on independent agents complicates the training and the transfer of knowledge. The exact quality measures are hand-tuned, and the agents designed by hand.

The advantage of the Matchbox architecture is that it is probabilistic in nature, making it possible to automatically learn the necessary models for any domain, and provide calibrated detection and localisation certainties. The design is also more transparent and easier to debug than the inherently non-deterministic multi-agent approach. It can also be more easily extended to include new patch-based and region-based methods, since many researchers already provide probabilistic models for their low-level algorithms.

The stochastic grammars [Zhu and Mumford, 2006] offer a very powerful mechanism for modelling images, both in terms of crisp constraints and probabilistic relations. The combination of logic and probabilities allows for correct modelling both at the level of the scene (where crisp constraints and partonomies play a large role), and the image level (where the inherently noisy data are best described in terms of probabilistic distributions). The ability to extend the grammar down to the image level provides a seamless bridge between high-level concepts and image primitives.

The major advantage of the Matchbox is that the appearance models are automatically learned. The And-Or graphs used to model scenes in Zhu's grammatical formalism must be constructed by hand, and have mostly been demonstrated on highly structured object classes. While there is considerable work going into constructing And-Or graphs for a large number of object categories, the extension of existing systems for handling thousands of categories of the visual world is likely to require learning. The probabilistic interface of the Matchbox allows for using off-the-shelf categorisation algorithms which have demonstrated good performance on more than a hundred categories.

The granularity of the middle-layer representation is another important difference between the Matchbox and some of the related approaches. Whereas both Zhu and Guhl use an abstraction continuum ranging from visual primitives (edgels, corners) to high-level categories and entire scenes, the Matchbox enforces a strict distinction between the sub-symbolic appearance model and the symbolic representation above the object level. Here it should be pointed out that the Matchbox can deal with many co-existing objects in the scene which form a partonomical hierarchy above the object level, in other words there is no reason why both an object and its parts (e.g. both a car and its wheels) can not be detected in the same image, using the high-level ontology and partonomical reasoning to resolve the apparent conflict. The sub-symbollic appearance model for each object category can also be hierarchical in nature, and the Matchbox is capable of passing high-level priors down to this level. The strict division between the object level and image level is less similar to a biological system than a smooth transition from image-level primitives to semantic categories, but it makes the architecture more generic, since it does not enforce a particular representation form on either the high-level or the low-level layers.

A further similarity to existing approaches is exhibited at the level of discretisation of feature dimensions. While any probabilistic classifier can be used to estimate the needed probabilities (in this work, SVMs and decision trees were used), the use of decision trees to estimate probabilities presents some additional possibilities. Since the decision trees presented in this thesis provide an axis-parallel discretisation of the feature space, each leaf of the tree represents a part of the feature space described as a combination of intervals. Such a description is easily represented and can be passed to a low-level segmentation algorithm as a generic form of feedback. This type of feedback is similar to that used by Coradeschi and Saffiotti [2000], who also use a combination of feature ranges to provide feedback to the tracking layer. The automatic learning of decision trees can also automate the process and learn a discretisation which attempts to optimise the classification rate. The discretisation of the feature space is also similar to the work on visual concept ontologies [Maillot and Thonnat, 2008], but the strength of the visual concept ontologies is that they have intuitive meaning that can be used by a human expert. The leaves of an automatically learned decision trees can be seen as an intermediate-level visual ontology of similar visual concepts, but such concepts are not necessarily intuitive and can not be used to communicate the visual knowledge to a human expert.

A number of object detection schemes have attempted to model complete scenes [Sudderth et al., 2005, Li et al., 2009]. While such work provides very useful tools for segmentation, object detection and annotation, it is limited to very simple object interactions. It does not seem likely that such methods will be capable of explaining complex scenes such as the one shown in Figure 1.1 in terms of intuitive, knowledge-based descriptions. The architecture proposed in this thesis therefore introduces a generic way to use context generated by a more complex reasoning system. Unlike the context classification method proposed by Rabinovich et al. [2007], the context integration formula presented in this work does not depend on a specific source of context (such as interaction potentials from an MRF) and the necessary conditions for the formula to hold are clearly defined.

The Matchbox is quite different from many robotics architectures. It does not perform fusion of information from different modalities, which is a very important feature for autonomous robots. Another important aspect is tracking in dynamic scenes, which was also not addressed. The reason for this is that single images offer sufficient information to be easily interpreted by humans, and are often the only thing available. While sensor fusion and tracking are very important problems, the question of how much we can understand from still images with state of the art algorithms is still a very interesting one.

## 6.4 Further Work

A number of low-level detectors have been integrated into the middle-layer architecture, but the most extensive integration was shown with the hierarchical patch-based method of Ommer and Buhmann [2010], where probabilistic high-level context can be used to improve the hypothesis generation, top-down grouping and the final classification. The region-based approaches described in this thesis were based on the assumption that a region covers most of the detected object and that the classification can be performed using a global feature vector extracted from the region's contents, which is not always a realistic assumption. However, there is recent reported work on object detection and modelling using hierarchical region representations, such as the work of Li et al. [2009], which allow for more accurate segmentation of the objects based on combining small regions. While there is no apparent reason against adapting such models to work with the proposed middle-layer, this integration has not been tested due to time constraints.

Another important aspect which needs to be addressed is the handling of occlusions and conflicts. The proposed architecture does provide a generic way of handling these issues, by using the partonomy and a special "can-overlap" relation in the knowledge base to allow certain overlaps and prohibit others. This approach is similar the one proposed by Nagao et al. [1979], and which was used in the early SIGMA system. Modern object detection approaches have demonstrated the ability to detect objects even in the presence of significant occlusion, so this form of reasoning can resolve the situations where the detections of two separate objects overlap. On the other hand, the categorisation methods based on shape models, which are commonly used for object detection, also offer more information than is currently used. Since they typically use generative models, they can also detect which parts of the object are occluded and which parts are clearly visible. The hierarchical approach of Ommer uses this information internally for selecting relevant compositions and top-down grouping, but this information is not explicitly used by the Matchbox to reason about occlusion.

Long-term extensions to the framework should certainly include the analysis of videos, which will bring additional considerations. Indexing mechanisms for efficient handling of data have already been implemented for static images, but the huge increase in the amount of data present in a typical video and potential real-time constraints will make these mechanisms much more important for the interpretation process. Also, the framework will have to track objects over many frames, bringing it closer to the work on perceptual anchoring. Another obvious extension is the introduction of multi-modal input, leading to sensor fusion. If the object detection remains probabilistic, this will require modelling of correlations between the data coming from different input channels.

Finally, more testing is required of the Matchbox as a part of a complete interpretation system. While the evaluation presented in this thesis shows that the architecture is capable of integrating high-level scene context into the object detection and classification framework, the effectiveness of a complete interpretation system built around the proposed middle layer has only been shown on a small number of images. A main problem is the lack of a large standardised annotated dataset geared specifically towards complete scene interpretation, similar to the standard benchmarks used by the object detection and categorisation communities. The annotated façade dataset presented in Chapter 5 is a step in this direction, but the complexity of the façade domain appears to have made the learning of good high-level models difficult at this time.

## 6.5 Outlook

Automatic understanding of complex images remains a difficult problem. This thesis has argued for a probabilistic model of the uncertainty at the level of object detections, which can be used for upward propagation of uncertainty and downward propagation of context. Both the object detection and the scene understanding communities have taken detections at the object level as a goal or starting point (depending on the perspective), so a unified representation of what has been observed in a scene has obvious advantages.

On the other hand, the recent developments in both image processing algorithms and computing power have brought about a number of interesting systems for understanding images and scenes. Such developments include improvements of traditional grammar-based and agent-based architectures, as well as novel probabilistic models of complete scenes. Some of them offer a more biologically inspired model of middle-level processing and show very promising early results. There is no consensus in the research community about the best way to model scenes, or the best way to detect and segment objects. This makes a generic architecture, such as the one presented in this thesis, very appealing since it allows for easier experimentation. On the other hand, a complete coherent model of a scene which reaches down to the image level, such as stochastic grammars or probabilistic models, might eventually result in better interpretations if the current problems related to learning and complexity are successfully addressed.

Despite exciting developments in recent years, the middle-layer problem has hardly been solved. It is the hope of the author that this work can contribute to further understanding of the problem and be a useful starting point for future development of image understanding systems.

Appendix A

# Estimating Evidence Likelihood using Decision Trees

This section describes the automatic learning of decision trees for estimating probability density in a multidimensional feature space. A decision tree is a tree where the leaf nodes represent classifications and each non-leaf node represents a decision rule acting on an attribute of the input sample. A sample described by a $d$-dimensional feature vector $f$ and consisting of $d$ scalar attributes is classified by evaluating the decision rule at the root node and passing the sample down to one of the subnodes depending on the result, until a leaf node is reached. The result is a partitioning of the feature space into labelled disjoint regions.

In a binary decision tree, the rules correspond to yes/no questions and each nonleaf node has exactly two children. The most common type of decision trees, called *univariate* decision trees, only act on one dimension at a time and thus result in an axis-parallel partitioning. The experiments described in this thesis used univariate decision trees.

If the leaves are allowed to correspond to more than one class, they are called impure leaves. If each class in a leaf node is given a probability, such trees can be used to model the uncertainty of the classification result. Essentially, they provide a discrete approximation of a probability density function, where the discretisation can be irregular.

## A.1 Learning algorithm

Since the space of all possible trees is huge, greedy algorithms for learning the best tree are usually employed. The tree starts as a single root node containing all the samples and is recursively subdivided according to a splitting criterion. There are many splitting criteria in use for learning decision trees, two of the most popular are information content and the Gini coefficient[1], used in our experiments. For both criteria, we need to know the conditional probability $P(\omega|t)$ at a node $t$, which can be approximated as

$$P(\omega_i|t) = \frac{N_i(t)}{N(t)}$$

where $N_i(t)$ is the number of samples in $t$ belonging to class $\omega_i$, and $N(t)$ is the number of all samples in $t$,

The information content $I(t)$ of a node $t$ is the number of bits needed to encode the information of that node. Given $m$ classes, the information content of an impure node is given as

$$I(t) = I(P(\omega_1|t), \ldots, P(\omega_m|t)) = \sum_{i=1}^{m} -P(\omega_i|t)log_2 P(\omega_i|t) \tag{A.1}$$

For a binary decision with equal probabilities, the information content is $I(\frac{1}{2}, \frac{1}{2}) = -\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2} = 1$bit. For a biased decision where one choice has a 99% probability, it is $I(\frac{1}{100}, \frac{99}{100}) = 0.08$ bits. The goal of decision tree learning is to minimise the uncertainty with every step, and since uncertain decisions are encoded with more information, splits which minimise the information content of the new trees should be favoured.

The Gini coefficient is a measure of the impurity of a node, and as such also related to the information content of the node. The Gini coefficient of node $t$ is defined as

$$G(t) = \sum_{i \neq j} P(\omega_i|t)P(\omega_j|t) \tag{A.2}$$

For a given split that divides $t$ into $t_l$ and $t_r$, the change in the Gini coefficient is given as

$$\Delta G(sp, t) = G(t) - (G(t_l)P_l + G(t_r)P_r)$$

where $P_l$ and $P_r$ are the priors for the left and right subnode, respectively. The best split is the one that maximises $\Delta G(sp, t)$.

When learning a decision tree, the node with the highest impurity (measured as high information content or high Gini coefficient) is split in a way that maximises the splitting criterion. This entails two decisions: choosing the attribute (dimension) to split on and choosing its best value for the split. A simple approach, used for learning the trees described in this thesis, is to perform an exhaustive search through the space of all possible splits in all possible dimensions, and to choose the one that minimises the Gini coefficient of the resulting sub-nodes. Given a set of $n$ samples, each described by a $d$-dimensional feature vector $f$, an exhaustive search of all possible splits in each dimension has a complexity of $O(nd)$.

[1]More detailed explanations can be found in [Poole et al., 1997] and [Webb, 2002]

## A.2 Pruning

Overfitting is a well-known problem with learning of decision trees [Poole et al., 1997, Webb, 2002]. The leaf-splitting can be continued until all leaves have pure class membership. Such learnt trees describe the training set well, but if the data are not perfectly separable or contain noise, they do not generalise well to unseen examples, essentially modelling the noise in the training set. The solution is to terminate the learning once a stopping criterion is fulfilled (e.g. minimum change of impurity function), or use one of many pruning algorithms to reduce the maximal tree.

Classification and Regression Trees (CART) were first introduced by Breiman [Breiman et al., 1984] and still present a popular method for pruning learnt decision trees. The basic idea is to add a constant $\alpha$ to the impurity measure at each split, as a measure of the cost of additional complexity introduced by the split.

More specifically, if $R(t)$ is the measure of impurity at node $t$ (the misclassification rate), then $R_\alpha(t) = R(t) + \alpha$ is the complexity measure of the node $t$. If $\tilde{T}$ is the set of all leaves in a tree $T$, and $|\tilde{T}|$ the cardinality of $\tilde{T}$, then $R(T) = \sum_{t \in \tilde{T}} R(t)$ is the estimated misclassification rate of a tree $T$, and

$$R_\alpha(T) = \sum_{t \in \tilde{T}} R_\alpha(t) = R(T) + \alpha|\tilde{T}|$$

is the estimated complexity-misclassification rate of $T$. If we define $T_t$ to be a subtree with node $t$ at its root, we can calculate the strength of the link from node $t$ to its leaves as

$$g(t) = \frac{R(t) - R(T_t)}{|\tilde{T}_t| - 1} \tag{A.3}$$

The nodes with a low $g(t)$ are punished as they add complexity without significantly improving the classification result. The algorithm starts with the maximal tree and calculates $g(t)$ for all nodes. The node with the lowest value of $g(t)$ is made into a leaf, and all of its children are removed. The new values for $g(t)$ are calculated for all the predecessors of the affected node, and the process is repeated on the new tree.

The result is a succession of trees, starting with the initial, maximal tree, and ending with a tree containing only the root node. Each of these trees is a classifier. All the trees are tested on an unseen validation dataset and the tree with the best classification rate is selected as the final classifier.

# Bibliography

S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *Proceedings of the 7th European Conference on Computer Vision*, pages 113–130, 2002.

S. Agarwal, A. Awan, and D. Roth. UIUC image database for car detection, 2002. URL http://l2r.cs.uiuc.edu/~cogcomp/Data/Car/. [Online; accessed 20-Jan-2010].

Y. Amit and D. Geman. A computational model for visual selection. *Neural Computation*, 11:1691–1715, 1998.

K. Andress and A. Kak. Evidence accumulation & flow of control in a hierarchical reasoning system. *AI Mag.*, 9(2):75–94, 1988. ISSN 0738-4602.

N. I. Badler. *Temporal Scene Analysis: Conceptual Descriptions of Object Movements.* PhD thesis, Dept. of Computer Science, University of Toronto, 1975.

L. R. Bahl, P. F. Brown, P. V. De, and R. L. Mercer. A tree-based statistical language model for natural language speech recognition. volume 37, pages 1001–1008, Jul 1989.

R. Bajcsy and J. Kosecka. The problem of signal and symbol integration: A study of cooperative mobile autonomous agent behaviors. In *In Proceedings of KI-95, LNCS*, pages 49–64. Springer, 1995.

D. H. Ballard, C. M. Brown, and J. A. Feldman. An approach to knowledge-directed image analysis. In *IJCAI'77: Proceedings of the 5th international joint conference on Artificial intelligence*, pages 664–670, Cambridge, USA, 1977. Morgan Kaufmann Publishers Inc.

M. Bar, K. S. Kassam, A. S. Ghuman, J. Boshyan, A. M. Schmid, A. M. Dale, M. S. Hamalainen, K. Marinkovic, D. L. Schacter, B. R. Rosen, and E. Halgren. Top-down facilitation of visual recognition. *Proceedings of the National Academy of Sciences of the United States of America*, 103(2):449–454, January 2006. ISSN 0027-8424. doi: 10.1073/pnas.0507062103. URL http://dx.doi.org/10.1073/pnas.0507062103.

I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987.

T. O. Binford. Bayesian inference in model-based machine vision. *Int. J. Approx. Reasoning*, 2(3):327–328, 1988.

H. Blum. Biological shape and visual science (part i). *Journal of Theoretical Biology*, 38(2):205–287, February 1973. ISSN 00225193. doi: 10.1016/0022-5193(73)90175-6. URL http://dx.doi.org/10.1016/0022-5193(73)90175-6.

W. Bohlken and B. Neumann. Generation of rules from ontologies for high-level scene interpretation. In *Proceedings of The International RuleML Symposium on Rule Interchange and Applications*, pages 93–107, Las Vegas, 2009. Springer.

E. Borenstein and S. Ullman. Combined top-down/bottom-up segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(12):2109–2125, 2008. ISSN 0162-8828. doi: http://dx.doi.org/10.1109/TPAMI.2007.70840.

G. Bouchard and B. Triggs. Hierarchical part-based visual object categorization. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 710–715, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2372-2. doi: http://dx.doi.org/10.1109/CVPR.2005.174.

M. Brand. Pattern discovery via entropy minimization. In *Artificial Intelligence and Statistics*. Morgan Kaufmann, 1998.

N. Bredeche, Y. Chevaleyre, J.-D. Zucker, A. Drogoul, and G. Sabah. A meta-learning approach to ground symbols from visual percepts. *Robotics and Autonomous Systems*, 43(2-3):149–162, 2003.

L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

W. Buntine. Learning classification trees. *Statistics and Computing*, 2:63–73, 1992.

M. C. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume II*, pages 628–641, London, UK, 1998. Springer-Verlag. ISBN 3-540-64613-2.

H. Buxton and S. Gong. Advanced visual surveillance using bayesian networks. In *International Conference on Computer Vision*, pages 111–123, 1995.

P. Carbonetto, N. de Freitas, and K. Barnard. A statistical model for general contextual object recognition. In *ECCV (1)*, pages 350–362, 2004.

A. Cavallaro and T. Ebrahimi. Interaction between high-level and low-level image analysis for semantic video object extraction. *EURASIP J. Appl. Signal Process.*, 2004: 786–797, 2004. ISSN 1110-8657. doi: http://dx.doi.org/10.1155/S1110865704402157.

J. Čech and R. Šára. Language of the structural models for constrained image segmentation. Technical Report Technical Report TN-eTRIMS-CMP-03-2007, Czech Technical University, Prague, 2007.

J. Čech and R. Šára. Languages for constrained binary segmentation based on maximum a posteriori probability labeling. *Int. J. Imaging Syst. Technol.*, 19(2):69–79, 2009. ISSN 0899-9457. doi: http://dx.doi.org/10.1002/ima.v19:2.

B. Cestnik. Estimating probabilities: A crucial task in machine learning. In *ECAI*, pages 147–149, 1990.

C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

A. Chella, M. Frixione, and S. Gaglio. A cognitive architecture for artificial vision. *Artif. Intell.*, 89(1-2):73–111, 1997. ISSN 0004-3702. doi: http://dx.doi.org/10.1016/S0004-3702(96)00039-2.

A. Chella, S. Coradeschi, M. Frixione, and A. Saffiotti. Perceptual anchoring via conceptual spaces. In *Proc. of the AAAI-04 Workshop on Anchoring Symbols to Sensor Data, AAAI*, 2004.

M. B. Clowes. On seeing things. *Artif. Intell.*, 2(1):79–116, 1971.

A. G. Cohn, D. R. Magee, A. Galata, D. C. Hogg, and S. M. Hazarika. Towards an architecture for cognitive vision using qualitative spario-temporal representations and abduction. In *In Spatial Cognition III*, pages 232–248. Springer-Verlag, 2002.

A. G. Cohn, D. C. Hogg, R. Möller, and B. Neumann, editors. *Logic and Probability for Scene Interpretation, Dagstuhl Seminar Proceedings 08091*, Dagstuhl, Feb 2008. URL `http://drops.dagstuhl.de/opus/portals/index.php?semnr=08091`.

T. F. Cootes and C. J. Taylor. Locating objects of varying shape using statistical feature detectors. In *ECCV (2)*, pages 465–474, 1996.

S. Coradeschi and A. Saffiotti. Anchoring symbols to sensor data: Preliminary report. In *AAAI/IAAI*, pages 129–135, 2000. URL `citeseer.ist.psu.edu/coradeschi00anchoring.html`.

K. Crammer, Y. Singer, N. Cristianini, J. Shawe-taylor, and B. Williamson. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:2001, 2001.

G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *In Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *In CVPR*, pages 886–893, 2005.

B. Draper, R. Collins, J. Brolio, A. Hanson, and E. Riseman. The schema system. Technical report, Amherst, MA, USA, 1988.

B. Draper, A. Hanson, and E. Riseman. Knowledge-directed vision: Control, learning, and integration. *PIEEE*, 84(11):1625–1637, November 1996.

M. Drauschke. An irregular pyramid for multi-scale analysis of objects and their parts. In *GbR'09, LNCS 5534*, pages 293–303, 2009.

M. Drauschke and W. Förstner. Comparison of adaboost and adtboost for feature subset selection. In *PRIS 2008*, Barcelona, Spain, 2008a.

M. Drauschke and W. Förstner. Selecting appropriate features for detecting buildings and building parts. In *21st Congress of the International Society for Photogrammetry and Remote Sensing (ISPRS)*, Beijing, China, 2008b.

K.-B. Duan and S. S. Keerthi. Which is the best multiclass svm method? an empirical study. pages 278–285. 2005. doi: 10.1007/11494683_28. URL http://dx.doi.org/10.1007/11494683_28.

B. Epshtein and S. Ullman. Feature hierarchies for object classification. In *ICCV*, pages 220–227, 2005.

M. Everingham. The VOC 2006 database, 2006. URL http://pascallin.ecs.soton.ac.uk/challenges/VOC/databases.html. [Online; accessed 20-Jan-2010].

S. E. Fahlman and G. E. Hinton. Connectionist architectures for artificial intelligence. *IEEE Computer*, 20(1):100–109, 1987.

L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 28:594–611, April 2006.

P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61:2005, 2003.

R. Fergus and P. Perona. The Caltech database, 2003. URL http://www.robots.ox.ac.uk/~vgg/data3.html. [Online; accessed 20-Jan-2010].

R. Fergus, P. Perona, and A. Zisserman. Weakly supervised scale-invariant learning of models for visual recognition. *Int. J. Comput. Vision*, 71(3):273–303, 2007. ISSN 0920-5691.

C. Fernández Tena, P. Baiget, X. Roca, and J. Gonzàlez. Natural language descriptions of human behavior from video sequences. In *KI '07: Proceedings of the 30th annual German conference on Advances in Artificial Intelligence*, pages 279–292, Osnabrück, Germany, 2007. Springer-Verlag. ISBN 978-3-540-74564-8. doi: http://dx.doi.org/10.1007/978-3-540-74565-5_22.

S. Fidler and A. Leonardis. Towards scalable representations of object categories: Learning a hierarchy of parts. In *CVPR*, 2007.

S. Fidler, D. Skocaj, and A. Leonardis. Combining reconstructive and discriminative subspace methods for robust classification and regression by subsampling. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(3):337–350, 2006. ISSN 0162-8828. doi: http://dx.doi.org/10.1109/TPAMI.2006.46.

R. Fikes, P. Hart, and N. Nilsson. Learning and executing generalized robot plans. Technical Report 70, AI Center, SRI International, 333 Ravenswood Ave, Menlo Park, CA 94025, Jul 1972. SRI Project 1530.

M. Fischler and R. Elschlager. The representation and matching of pictorial structures. 22(1):67–92, January 1973.

K. S. Fu. *Syntactic Pattern Recognition and Applications*. Prentice Hall, 1982.

K. Fukushima. Neocognitron for handwritten digit recognition. *Neurocomputing*, 51: 161–180, 2003.

F. Fusier, V. Valentin, F. Bremond, M. Thonnat, M. Borg, D. Thirde, and J. Ferryman. Video understanding for complex activity recognition. *Machine Vision and Applications (MVA)*, 18:167–188, August 2007.

A. Galata, A. Cohn, D. Magee, and D. Hogg. Modeling interaction using learnt qualitative spatio-temporal relations and variable length markov models. In *In Proceedings of the European Conference on Artificial Intelligence*, pages 741–745, 2002.

P. Gärdenfors. *Conceptual Spaces: The Geometry of Thought*. The MIT Press, March 2000.

R. Gerber, H.-H. Nagel, and H. Schreiber. Deriving textual descriptions of road traffic queues from video sequences. In *in Proc. 15th European Conference on Artificial Intelligence (ECAI-2002)*, pages 736–740. IOS Press, 2002.

J. J. Gibson. *The Theory of Affordances*. Lawrence Erlbaum, 1977.

G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007. URL http://authors.library.caltech.edu/7694.

C. Gu, J. Lim, P. Arbelaez, and J. Malik. Recognition using regions. In *CVPR09*, pages 1030–1037, 2009.

T. P. Guhl. *Machine Perception using a Blackboard Architecture*. PhD thesis, 2007.

T. P. Guhl and M. P. Shanahan. Machine perception using a blackboard architecture. In *The 5th International Conference on Computer Vision Systems*, 2007.

C.-e. Guo, S.-C. Zhu, and Y. N. Wu. Primal sketch: Integrating structure and texture. *Comput. Vis. Image Underst.*, 106(1):5–19, 2007. ISSN 1077-3142. doi: http://dx.doi.org/10.1016/j.cviu.2005.09.004.

E. Gyftodimos and P. A. Flach. Hierarchical bayesian networks: A probabilistic resoning model for structured domains. In E. de Jong and T. Oates, editors, *Proc. Workshop on Development of Representations*, ICML, pages 23–30, 2002.

A. Hanson and E. Riseman. Visions: A computer system for interpreting scenes. *Computer Vision Systems*, pages 303–333, 1978.

S. Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42:335–346, 1990. doi: 10.1016/0167-2789(90)90087-6. URL http://www.isrl.uiuc.edu/~amag/langev/paper/harnad90theSymbol.html.

J. Hartz and B. Neumann. Learning a knowledge base of ontological concepts for high-level scene interpretation. In *IEEE Proc. International Conference on Machine Learning and Applications*, Cincinnati (Ohio, USA), Dec 2007.

J. Hartz, L. Hotz, B. Neumann, and K. Terzić. Automatic incremental model learning for scene interpretation. In *to appear in: Proceedings of the IASTED International Conference on Computational Intelligence*, Honolulu, Hawaii, August 2009.

J. Hartz, A. Koopmann, P. Kreutzmann, and K. Terzic. eTRIMS scene interpretation datasets. Technical Report Technical Report FBI-HH-M-345/10, Universität Hamburg, November 2010.

T. Hastie and R. Tibshirani. Classification by pairwise coupling. In *NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10*, pages 507–513, Cambridge, MA, USA, 1998. MIT Press. ISBN 0-262-10076-2.

D. Heesch and M. Petrou. Markov random fields with asymmetric interactions for modelling spatial context in structured scenes. *Journal of Signal Processing Systems, to appear*, 2009.

F. Heintz and P. Doherty. DyKnow: An approach to middleware for knowledge processing. *Journal of Intelligent and Fuzzy Systems*, 15(1):3–13, nov 2004.

F. Heintz and P. Doherty. DyKnow federations: Distributing and merging information among UAVs. In *Proceedings of the Eleventh International Conference on Information Fusion (Fusion'08)*, 2008. URL http://www.ida.liu.se/~frehe/publications/fusion2008.pdf. June 30-July 3, Cologne, Germany.

F. Heintz, P. Rudol, and P. Doherty. Bridging the sense-reasoning gap using dyknow: A knowledge processing middleware framework. In J. Hertzberg, M. Beetz, and R. Englert, editors, *KI 2007: Advances in Artificial Intelligence*, volume 4667, pages 460–463. Springer, Berlin Heidelberg, 2007. URL http://www.ida.liu.se/~frehe/publications/ki2007.pdf.

H. Hexmoor, J. Lammens, and S. C. Shapiro. Embodiment in glair: a grounded layered architecture with integrated reasoning for autonomous agents. In D. D. D. II and J. Stewman, editors, *Proceedings of The Sixth Florida AI Research Symposium (FLAIRS 93)*, pages 325–329. Florida AI Research Society, April 1993.

S. Hongeng, F. Bremond, and R. Nevatia. Representation and optimal recognition of human activities. In *IEEE Proceedings of Computer Vision and Pattern Recognition (CVPR'00)*, South Carolina, USA, 2000.

L. Hotz and B. Neumann. Scene Interpretation as a Configuration Task. *Künstliche Intelligenz*, 3:59–65, 2005.

L. Hotz, B. Neumann, K. Terzić, and J. Šochman. Feedback between low-level and high-level image processing. Technical Report Report FBI-HH-B-278/07, Universität Hamburg, Hamburg, 2007.

L. Hotz, B. Neumann, and K. Terzic. High-level expectations for low-level image processing. In *KI 2008: Advances in Artificial Intelligence*, volume 5243 of *Springer Lecture Notes in Computer Science*, pages 87–94, 2008.

C.-W. Hsu and C.-J. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=991427&isnumber=21380.

B. Hummel, W. Thiemann, and I. Lulcheva. Scene understanding of urban road intersections with description logic. In A. G. Cohn, D. C. Hogg, R. Möller, and B. Neumann, editors, *Logic and Probability for Scene Interpretation*, number 08091 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2008. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany. URL http://drops.dagstuhl.de/opus/volltexte/2008/1616.

D. Hutber, S. Moisan, C. Shekhar, and M. Thonnat. Perception-interpretation interfacing for the prolab2 road vehicle. In *7th Symposium on Transportation Systems: Theory and Application of Advanced Technology*, Tianjin, China, Aug 1994.

N. D. Huu, W. Paquier, and R. Chatila. Combining structural descriptions and image-based representations for image, object, and scene recognition. In *IJCAI*, pages 1452–1457, 2005.

M. Jahangiri and M. Petrou. Fully bottom-up blob extraction in building facades. In *PRIA*, 2008.

F. V. Jensen, H. I. Christensen, and J. Nielsen. Bayesian methods for interpretation and control in multi-agent vision systems. In *Applications of Artificial Intelligence X: Machine Vision and Robotics*, volume 1708 of *SPIE Proceedings Series*, 1992.

Y. Jin and S. Geman. Context and hierarchy in a probabilistic image model. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:2145–2152, 2006. ISSN 1063-6919. doi: http://doi.ieeecomputersociety.org/10.1109/CVPR.2006.86.

T. Joachims. Text categorization with suport vector machines: Learning with many relevant features. In *ECML*, pages 137–142, 1998.

T. Joachims. Making large-scale support vector machine learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in kernel methods: support vector learning*, pages 169–184. MIT Press, Cambridge, MA, USA, 1999. ISBN 0262194163.

B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290(5802):91–97, March 1981. doi: 10.1038/290091a0.

T. Kadir and M. Brady. Saliency, scale and image description. *Int. J. Comput. Vision*, 45 (2):83–105, 2001. ISSN 0920-5691. doi: http://dx.doi.org/10.1023/A:1012460413855.

Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *CVPR*, pages 506–513, 2004.

B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker. Shapes, shocks, and deformations i: The components of two-dimensional shape and the reaction-diffusion space. *International Journal of Computer Vision*, 15:189–224, 1994.

R. Kirsch. Computer interpretation of english text and picture patterns. *IEEE Transactions on Electronic Computers*, 13:363–376, Aug 1964.

C. A. Kohl, A. R. Hanson, and E. M. Riseman. A goal-directed intermediate level executive for image interpretation. In *IJCAI'87: Proceedings of the 10th international joint conference on Artificial intelligence*, pages 811–814, Milan, Italy, 1987. Morgan Kaufmann Publishers Inc.

A. Kojima, T. Tamura, and K. Fukunaga. Natural language description of human activities from video images based on concept hierarchy of actions. *International Journal of Computer Vision*, 50:171–184, 2002.

D. Koller and A. Pfeffer. Object-oriented bayesian networks. In *The Thirteenth Annual Conference on Uncertainty in Artificial Intelligence*, pages 302–313, August 1997.

K. Konolige, K. L. Myers, E. H. Ruspini, and A. Saffiotti. The saphira architecture: a design for autonomy. *J. Exp. Theor. Artif. Intell.*, 9(2-3):215–235, 1997.

F. Korč and D. Schneider. Annotation tool. Technical Report TR-IGG-P-2007-01, June 2007. URL http://www.ipb.uni-bonn.de/html_pages_software/annotation-tool/publ/Korc-TR-IGG-P-2007-01.pdf.

F. Korč and W. Förstner. Interpreting terrestrial images of urban scenes using discriminative random fields. In *Proc. of the 21st Congress of the International Society for Photogrammetry and Remote Sensing (ISPRS)*, 2008.

S. Krempp, D. Geman, and Y. Amit. Sequential learning of reusable parts for object detection. Technical report, CS John Hopkins, 2002.

A. Kreutzmann, K. Terzić, and B. Neumann. Context-aware classification for incremental scene interpretation. In *Workshop on Use of Context in Vision Processing*, Boston, November 2009.

M. Lades, J. C. Vorbrüggen, J. Buhmann, J. Lange, C. V. D. Malsburg, R. P. Würtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Trans. Computers*, 42:300–311, 1993.

S. Lazebnik, C. Schmid, and J. Ponce. Semi-local affine parts for object recognition. In *In BMVC*, pages 959–968, 2004.

B. Leibe and B. Schiele. Interleaved object categorization and segmentation. In *In BMVC*, pages 759–768, 2003.

B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Proceedings of the Workshop on Statistical Learning in Computer Vision*, Prague, Czech Republic, May 2004.

B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 878–885 vol. 1, 2005.

M. Leyton. A process-grammar for shape. *Artif. Intell.*, 34(2):213–247, 1988. ISSN 0004-3702. doi: http://dx.doi.org/10.1016/0004-3702(88)90039-2.

L.-J. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *Computer Vision and Pattern Recognition (CVPR-09)*, 2009.

S. Z. Li. Markov random field models in computer vision. In *ECCV (2)*, pages 361–370, 1994.

N. Logothetis and D. Sheinberg. Visual object recognition. *Annu. Rev. Neurosci.*, 19: 577–621, 1996.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

N. Maillot and M. Thonnat. Ontology based complex object recognition. *Image Vision Comput.*, 26(1):102–113, 2008. ISSN 0262-8856. doi: http://dx.doi.org/10.1016/j.imavis.2005.07.027.

N. Maillot, M. Thonnat, and A. Boucher. Towards ontology based cognitive vision. In *ICVS'03: Proceedings of the 3rd international conference on Computer vision systems*, pages 44–53, Berlin, Heidelberg, 2003. Springer-Verlag. ISBN 3-540-00921-3.

N. Maillot, M. Thonnat, and A. Boucher. Towards ontology-based cognitive vision. *Mach. Vision Appl.*, 16(1):33–40, 2004. ISSN 0932-8092. doi: http://dx.doi.org/10.1007/s00138-004-0142-9.

D. Marr. *Vision*. Freeman Publisher, 1983.

M. Marszalek, I. Laptev, and C. Schmid. Actions in context. In *CVPR*, pages 2929–2936, 2009.

T. Matsuyama and V. S. Hwang. *SIGMA: A Knowledge-Based Aerial Image Understanding System*. Perseus Publishing, 1990. ISBN 030643301X.

T. Matsuyama and V. S.-S. Hwang. Sigma: A framework for image understanding - integration of bottom-up and top-down analysis. In *IJCAI*, pages 908–915, 1985.

D. M. McKeown, W. A. Harvey, and J. McDermott. Rule-based interpretation of aerial imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7:570–585, Sep 1985.

H. Meine. *The GeoMap Representation: On Topologically Correct Sub-pixel Image Analysis*. PhD thesis, Department Informatik, Universität Hamburg, jan 2009.

K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(10):1615–1630, 2005. ISSN 0162-8828.

J. W. Modestino and J. Zhang. A markov random field model-based approach to image interpretation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(6):606–615, 1992.

M. Mohnhaupt and B. Neumann. Understanding object motion: recognition, learning and spatiotemporal reasoning. pages 65–91, 1993.

S. Muggleton and L. de Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19:629–679, 1994.

M. Nagao, T. Matsuyama, and H. Mori. Structural analysis of complex aerial photographs. In *IJCAI'79: Proceedings of the 6th international joint conference on Artificial intelligence*, pages 610–616, San Francisco, CA, USA, 1979. Morgan Kaufmann Publishers Inc. ISBN 0-934613-47-8.

H.-H. Nagel. Analyzing sequences of tv-frames. In *Proceedings of the 5th International Join t Conference on Artificial Intelligence*, pages 22–25, 1977.

H.-H. Nagel. From image sequences towards conceptual descriptions. *Image and Vision Computing*, 6(2):59–74, 1988.

H.-H. Nagel. Steps toward a cognitive vision system. *AI Magazine*, 25(2):31–50, 2004. ISSN 0738-4602.

R. Narasimhan. Syntax-directed interpretation of classes of pictures. *Commun. ACM*, 9 (3):166–173, 1966. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/365230.365258.

B. Neumann. A conceptual framework for high-level vision. Technical Report FBI-HH-B245/02, FB Informatik, Universität Hamburg, Hamburg, July 2002.

B. Neumann. Bayesian compositional hierarchies - a probabilistic structure for scene interpretation. Technical Report FBI-HH-B-282/08, Universität Hamburg, Department Informatik, Arbeitsbereich Kognitive Systeme, May 2008.

B. Neumann and R. Möller. On scene interpretation with description logics. *Image and Vision Computing*, 26(1):82–101, 2008. ISSN 0262-8856.

B. Neumann and H.-J. Novak. Naos: Ein system zur natürlichsprachlichen beschreibung zeitveränderlicher szenen. *Inform., Forsch. Entwickl.*, 1(2):83–92, 1986.

B. Neumann and K. Terzić. Context-based probabilistic scene interpretation. In *Proceedings of the Third IFIP International Conference on Artificial Intelligence in Theory and Practice*, Brisbane, Australia, Sep 2010.

H. P. Nii, E. A. Feigenbaum, J. J. Anton, and A. J. Rockmore. Signal-to-symbol transformation: Hasp/siap case study. *AI Magazine*, 3(2):23–35, 1982.

N. J. Nilsson. Shakey the robot. Technical Report 323, AI Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, Apr 1984.

E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *In Proc. ECCV*, pages 490–503. Springer, 2006.

Y. Ohta. *Knowledge-based interpretation of outdoor natural color scenes.* Pitman Publishing, Inc., Marshfield, MA, USA, 1985. ISBN 0-273-08673-1.

B. Ommer and J. Buhmann. Learning the compositional nature of visual object categories for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):501–516, 2010. ISSN 0162-8828. doi: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2009.22.

A. Opelt, A. Pinz, M. Fussenegger, and P. Auer. Generic object recognition with boosting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(3):416–431, 2006. ISSN 0162-8828. doi: http://dx.doi.org/10.1109/TPAMI.2006.54.

S. E. Palmer. Hierarchical structure in perceptual representation. *Cognitive Psychology*, 9(4):441–474, October 1977.

S. E. Palmer. *Vision science : photons to phenomenology.* MIT Press, Cambridge, Mass., 1999.

C. P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, page 555, Washington, DC, USA, 1998. IEEE Computer Society. ISBN 81-7319-221-9.

J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

J. C. Platt, N. Cristianini, and J. Shawe-taylor. Large margin dags for multiclass classification. In *Advances in Neural Information Processing Systems*, pages 547–553. MIT Press, 2000.

D. Poole, A. Mackworth, and R. Goebel. *Computational intelligence: a logical approach.* Oxford University Press, Oxford, UK, 1997. ISBN 0-19-510270-3.

A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *ICCV*, pages 1–8, 2007.

Randell, D.A., Z. Cui, and A. G. Cohn. A spatial logic based on regions and connection. In *Proceedings 3rd International Conference on Knowledge Representation and Reasoning*, 1992.

J. T. Rickard, J. Aisbett, and G. Gibbon. Reformulation of the theory of conceptual spaces. *Inf. Sci.*, 177(21):4539–4565, 2007. ISSN 0020-0255.

R. D. Rimey. Control of selective perception using bayes nets and decision theory. Technical Report 468, University of Rochester, Computer Science Department, Rochester, New York 14627, Dec 1993.

R. D. Rimey and C. M. Brown. Control of selective perception using bayes nets and decision theory. *Int. J. Comput. Vision*, 12(2-3):173–207, 1994. ISSN 0920-5691. doi: http://dx.doi.org/10.1007/BF01421202.

N. Ripperda and C. Brenner. Evaluation of structure recognition using labelled facade images. In *Proceedings of the 31st DAGM Symposium on Pattern Recognition*, pages 532–541, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-03797-9. doi: http://dx.doi.org/10.1007/978-3-642-03798-6_54.

C. A. Rosen and N. J. Nilsson. Application of intelligent automata to reconnaissance. Technical report, Stanford Research Institute, November 1966. Project 5953 Interim Report 1 From the Nilsson archives SHAKEY papers.

H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions On Pattern Analysis and Machine intelligence*, 20:23–38, 1996.

B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation. Technical report, Tech. Rep. MIT-CSAIL-TR-2005-056, Massachusetts Institute of Technology, 2005.

F. Scalzo and J. H. Piater. Statistical learning of visual feature hierarchies. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, page 44, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2372-2-3. doi: http://dx.doi.org/10.1109/CVPR.2005.532.

C. Schlegel, J. Illmann, H. Jaberg, M. Schuster, and R. Wörz. Integrating vision based behaviours with an autonomous robot. In *ICVS '99: Proceedings of the First International Conference on Computer Vision Systems*, pages 1–20, London, UK, 1999. Springer-Verlag. ISBN 3-540-65459-3.

C. Schmid. Constructing models for content-based image retrieval. In *In Proceedings CVPR*, pages 39–45, 2001.

Y.-W. Seo, N. Ratliff, and C. Urmson. Self-supervised aerial image analysis for extracting parking lot structure. In *Proc. of Twenty-First Int. Joint Conf. on AI IJCAI-09*, pages 1837–1842, Pasadena, 2009.

M. Shanahan. Prediction is deduction but explanation is abduction. In *Proceedings IJCAI 89*, pages 1055–1060. Morgan Kaufmann, 1989.

M. Shanahan. Perception as abduction: Turning sensor data into meaningful representation. *Cognitive Science*, 29:103–134, 2005.

S. C. Shapiro and H. O. Ismail. Anchoring in a grounded layered architecture with integrated reasoning. *Robotics and Autonomous Systems*, 43(2-3):97 – 108, 2003. Perceptual Anchoring: Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems.

V. Shet, D. Harwood, and L. Davis. Vidmap: Video monitoring of activity with prolog. In *IEEE International Conference on Advanced Video and Signal based Surveillance*, pages 224–229, 2005.

J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *Int. J. Comput. Vision*, 81(1):2–23, 2009. ISSN 0920-5691. doi: http://dx.doi.org/10.1007/s11263-007-0109-1.

J. Šochman and J. Matas. Learning fast emulators of binary decision processes. *International Journal of Computer Vision*, 83(2):149–163, June 2009. doi: 10.1007/s11263-009-0229-x.

V.-W. Soo, C.-Y. Lee, J. J. Yeh, and C.-c. Chen. Using sharable ontology to retrieve historical images. In *JCDL '02: Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries*, pages 197–198, Portland, Oregon, USA, 2002. ACM. ISBN 1-58113-513-0. doi: http://doi.acm.org/10.1145/544220.544261.

E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *In IEEE Intl. Conf. on Computer Vision*, pages 1331–1338, 2005.

K. K. Sung and T. Poggio. Example based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:39–51, 1995.

M. Taddeo and L. Floridi. Solving the symbol grounding problem: a critical review of fifteen years of research. *J. Exp. Theor. Artif. Intell.*, 17(4):419–445, 2005.

K. Terzić and B. Neumann. Decision trees for probabilistic top-down and bottom-up integration. Technical Report Technical Report FBI-HH-B-288/09, Universität Hamburg, Hamburg, 2009a.

K. Terzić and B. Neumann. Integrating context priors into a decision tree classification scheme. In *International Conference on Machine Vision, Image Processing, and Pattern Analysis*, Bangkok, Thailand, 2009b.

K. Terzić, L. Hotz, and B. Neumann. Division of work during behaviour recognition - the SCENIC approach. In *Workshop on Behaviour Modelling and Interpretation, 30th German Conference on Artificial Intelligence*, Osnabrück, Germany, September 2007.

K. Terzić, L. Hotz, and J. Šochman. Interpreting structures in man-made scenes: Combining low-level and high-level structure sources. In *International Conference on Agents and Artificial Intelligence*, Valencia, Spain, Jan 2010.

J. Thureson and S. Carlsson. Appearance based qualitative image description for object class recognition. In *In Proc. ECCV*, pages 518–529, 2004.

S. Todorovic and N. Ahuja. Unsupervised category modeling, recognition, and segmentation in images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(12):2158–2174, 2008.

A. Torralba, K. P. Murphy, and W. T. Freeman. The MIT-CSAIL database of objects and scenes, 2003. URL http://web.mit.edu/torralba/www/database.html. [Online; accessed 20-Jan-2010].

A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *In CVPR*, pages 762–769, 2004.

J. K. Tsotsos. Analyzing vision at the complexity level. *The behavioral and brain science*, 13:423–469, 1990.

S. Ullman. Visual routines. *Cognition*, 18:97–159, 1984.

V. N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag New York, Inc., New York, NY, USA, 1995. ISBN 0-387-94559-8.

V. N. Vapnik. *Statistical Learning Theory.* Wiley-Interscience, 1998.

P. Viola and M. J. Jones. Robust real-time face detection. *Int. J. Comput. Vision*, 57(2): 137–154, 2004. ISSN 0920-5691. doi: http://dx.doi.org/10.1023/B:VISI.0000013087.49260.fb.

E. Wachsmuth, M. Oram, and D. Perrett. Recognition of objects and their component parts: Responses of single units in the temporal cortex of the macaque. *Cerebral Cortex*, 4:509–522, 1994.

G. Wahba. Support vector machines, reproducing kernel hilbert spaces and the randomized gacv. In B.Schölkopf, C.J.C.Burges, and A.J.Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 69–88, Cambridge, MA, 1999. MIT Press.

A. R. Webb. *Statistical Pattern Recognition, 2nd Edition.* John Wiley & Sons, October 2002.

M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part I*, pages 18–32, London, UK, 2000. Springer-Verlag. ISBN 3-540-67685-6.

T.-F. Wu, C.-J. Lin, and R. C. Weng. Probability estimates for multi-class classification by pairwise coupling. *J. Mach. Learn. Res.*, 5:975–1005, 2004. ISSN 1532-4435.

A. L. Yuille. Deformable templates for face recognition. *J. Cognitive Neuroscience*, 3 (1):59–70, 1991. ISSN 0898-929X.

B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *In Proceedings of the Eighteenth International Conference on Machine Learning*, pages 609–616. Morgan Kaufmann, 2001.

H. Zhang, W. Jia, X. He, and Q. Wu. Learning-based license plate detection using global and local features. *Pattern Recognition, International Conference on*, 2:1102–1105, 2006. ISSN 1051-4651. doi: http://doi.ieeecomputersociety.org/10.1109/ICPR.2006.758.

L. Zhu, A. Rao, and A. Zhang. Theory of keyblock-based image retrieval. *ACM Trans. Inf. Syst*, 20:224–257, 2002.

S. Zhu and D. Mumford. *A Stochastic Grammar of Images.* Foundations and Trends in Computer Graphics and Vision. Prentice-Hall, 2006.