Situated Computer Vision

Sven Wachsmuth

Habilitationsschrift
(Tag der Habilitation: 30.1.2009)

Universität Bielefeld
Technische Fakultät

March 24, 2010

Contents

Preface

We are currently witnessing a dramatic change of the kind and manner how people interact with computing machinery. Although most people still use keyboard and mouse for their desktop personal computers, more and more computational units invade our daily life that cannot be easily accessed by traditional means of human-computer interfaces. The interaction space grows from the easily controlled virtual desktop to an uncontrolled physical environment which is the domain of natural human-human communication. One reason for this is *miniaturization* like cell phones which are becoming more and more computationally powerful. Another reason is distribution or *pervasiveness*. Computational units are integrated everywhere in our environment without being noticed. There is no physical instance for plugging in a monitor or a keyboard. A third reason is *embodiment*. The appearance and movement of robotic toys, like the Sony AIBO, of robotic interfacing agents, like the Philips iCat, or even of humanoid robots, like the Honda Asimo, mimic human-like or animal-like characters. They are situated in the physical world and not in a digital world. Their embodiment and character-style causes significant degrees of anthropomorphism, i.e. the attribution of uniquely human characteristics and qualities. An important part of it is the expectation to communicate with these technical platforms in a human-style fashion. A fourth reason is the availability of *technology*. Many of the utilities mentioned have built-in cameras and microphones which resembles the most important and richest sensory modalities of humans. As a consequence, there is an economic and social pressure to use them and to produce devices that are more fun to interact with.

Human-human communication has many facets. It is not only based on language but an inherently multi-modal affair that involves every sense that we have. The sender as well as the receiver makes extensive use of them in coding as well as decoding a communicative goal or intention. This can be

verified in everyone's own personal experience. We even do it when the communicative channel does not transmit the multi-modal content which might cause some irritation on the receiver side. For example, a person presenting a talk via laptop and beamer is pointing on his computer screen although nobody can see it, a young child telephoning with her Grandma shows her newest toy when asked about her birthday presents although Grandma can only hear her, two persons communicating through a closed window are verbally commenting what they show to each other although nobody can hear anything. It looks irritating because the different multi-modal cues that are produced by the actors relate and reference to each other. They are *situated* in that they do not encode the full meaning. Other cues of the current situation are needed in order to complete their understanding.

This does not only account for communicative situations, but is a more general principle. Many computer vision techniques seem to be fragile when they are taken out only slightly of the application scenario they are designed for. This is an inherent problem that has already been noticed a long time ago. Solutions can be based on different principles. First, we can add explicit *contextual knowledge* to the system that is used to control the application of image operators. Secondly, we can add *contextual features* that influence or bias the classification decision of some interpretation process. Thirdly, we can actively *shape the capturing process* in order to make the interpretation process more invariant to context. Fourthly, we can provide appropriate *feedback* about the current system performance so that a potential user can *change the current situation* for a more proper system performance. All these strategies are different variations of *situated computer vision* approaches. They become especially important if computer vision results need to be communicated to a human user. In this case, not all computer vision results matter, only some selective aspects of a scene are of interest. For a complete interpretation, these need to be related to the user's expectations. Thus, the visual interpretation process becomes embedded in a kind of user-system dialog that can be shaped by verbal statements as well as various other non-verbal contextual cues. One example is joint attention that leads to coupled capturing processes of communication partners. Another example is prompting the user with computer vision results. This establishes feedback loops that give an idea of a successful or unsuccessful information exchange.

The modeling of computer vision as a situated process is the general topic of this thesis. First, we will relate it to general trends in the computer vision community and to what has been found for the human perceptual system.

Psychological experiments have shown that context is extensively used in the human brain. Visual understanding and language production/perception influence each other on a very early stage of processing. These aspects are discussed in **Chapter 1**.

Chapter 2 takes a more technical standpoint and discusses several techniques for considering static scenes and their relations to objects. This is continued in **Chapter 3** for dynamic scenes. The interpretation of human actions inherently involves context because its physical performance is directed towards an environmental state change.

Situations group previously unrelated items into a coherent context. Interpretation processes can exploit this grouping by making relations between these items explicit. However, we can also focus on the dual process which infers the relations from many occurrences of situational groupings. Thus, context is exploited for model acquisition and learning. Semantic relations between words and visual items is a typical example and documents containing both are omnipresent, e.g., in the world wide web. This topic is treated in **Chapter 4**.

Frequently, context and situativity is also a matter of control. A system that is embedded in the physical world continuously needs to react on changing environmental conditions. It needs to take decisions that irreversibly change its own environment. An optimal decision depends on the current situation and might lead to a new situation. These aspects are treated in **Chapter 5**. However, systems that perceive and interact with their environment are too complex to be monolithic. They need to deal with many things in parallel. Typically, the control is distributed over several components. Although, several frameworks have been proposed that simplify the component-based construction of larger systems, the system integration task is frequently underestimated. It involves more aspects than only control. **Chapter 6** discusses different principles and frameworks that keep situated systems manageable.

Chapter 1

Situated Perception

In this chapter, I introduce the term *situated computer vision* and discuss some related empirical findings of human perception. The term resembles many ideas that already have been established in computer vision research. *Situatedness* refers to an inherent *ambiguity* occurring in a *selective* perception process. The perceiver needs to be aware of the current situation in order to infer an intended result because interpretations are uncertain and different interpretations might be possible. Such situational constraints can be formulated as contextual knowledge or as a *prior* of a probability distribution. Perception is modeled as a selective process, i.e. it does not aim at a complete nor generic object or scene understanding. Instead, perception is embedded in a purposive capturing process or a related task. It could also be constraint by the system's own embodiment as suggested by recent work in cognitive systems. Thus, there is a large spectrum of possible situational constraints ranging from physical embodiment to mental models. The latter have been introduced – as *situation models* – already a long time ago in research on human text comprehension. Situation models have been first described as amodal representations, i.e. independent of any perceptual process. However, newer experimental studies show that they are tightly linked to visual perception. Overall, context seems to have a fundamental role in the human visual pathway. This should be even more the case if vision takes place in the context of communicative situations.

1.1 Perspectives on computer vision

Computer vision is a heterogeneous field that embraces a large spectrum of methods as well as scientific perspectives. This starts with the physical understanding of the plenoptic function that describes how the light gets refracted, reflected, scattered, or absorbed with regard to a scene. In this terms computer vision could be understood as the inverse function of computer graphics reconstructing a scene from the illumination measured by a camera. A second perspective on computer vision is to mimic biological vision in order to get a deeper understanding of involved processes, representations, and architectures. Here, it is becoming more and more obvious that the fundamental questions and open problems in computer vision are at the cutting edge of cognition research. They cannot be solved in isolation but concern the fundamental basis of cognition itself. A third perspective understands computer vision as an engineering discipline that aims at the solution of practical vision tasks. But instead of a systematic methodological approach, the current state-of-the-art is mainly dominated by heuristics and knowledge from experience. All three perspectives cannot be separated and deeply influence each other which – together with the technical progress – has made computer vision a highly dynamic field over the last 50 years.

Computer vision as a reasoning problem

The early roots of computer vision started in two different research fields – artificial intelligence (AI) and pattern recognition. With the foundation of AI at the Dartmouth conference 1956 also vision was defined as a sub-area for applying AI techniques. In Newell's and Simon's ideas of a general problem solver they assume separate sensory systems for different kinds of information about the external environment. A computer vision sub-system needs to extract symbolic representations from image data that binds physical events to the internal knowledge base. In this kind of tradition, Ballard and Brown defined a computer vision task as the "construction of explicit meaningful descriptions of physical objects from images" (Ballard & Brown, 1982).

In order to simplify this task intermediate representation levels (generalized images, segmented images, geometric representations, relational models) are introduced that enable the system to apply different kinds of constraints and rules that limit the search space of possible interpretations. In determining the constraints and rules on each level, the design of computer vision

systems can be viewed as a knowledge engineering task. How to deal with dynamic bottom-up and top-down restrictions in system control can still be learned from these approaches. In his influential book Marr (1982) introduces different abstraction levels that guide the processing from image representations via primal sketch, $2\frac{1}{2}$D sketch to a structured and object-centered 3D model. As also argued by Jackendoff (1987) this 3D level would be the appropriate level for conceptual binding and translation to other modalities like language. In the view of Marr, vision is seen as a grouping and reconstruction process of 3D shape. A key aspect that still keeps the basic ideas of David Marr attractive is the generic formulation of the visual interpretation process. The problem of generic object recognition is still unsolved and – due to some researchers' perspective – today's most established computer vision techniques are even farther from solving it.

Computer vision as an estimation problem

Besides the influence of AI, pattern recognition techniques played a thorough rule from the beginnings of computer vision (Duda & Hart, 1973; Schürmann, 1977). Here, the sensor data – the signal – is assumed to be generated by a stochastic process. Classifiers are designed in order to reverse the stochastic process. Given a pattern representation of the sensor data the classifier assigns a class label that refers to the original signal source. Typically the classifier is learned from representative training samples. The design of a classifier is based on an appropriate feature selection and an appropriate modeling of the decision function. In an extreme case, the pixel representation of an image itself could define a feature vector. This opens up the possibility of a shortcut in visual processing. Instead of reconstructing an object in 3D, first, and then map the object-centered representation to a semantic meaning, the 2D image data can be directly linked to semantic object labels. Such kind of appearance-based techniques became popular in the early 90s (Turk & Pentland, 1991) and are still an area of active research (Viola & Jones, 2004; Lowe, 1999)

Computer vision as a control problem

Starting around 1985, ideas from another cognitive theory deeply influenced the field of computer vision. Ruzena Bajcsy argued that "the problem of perception was not necessarily one of signal processing but of control of data

acquisition" (Bajcsy, 1985, 1988). Bajcsy's work was motivated from the ecological approach to perception as formulated by Gibson (1950). Gibson put forward that perception does not aim at generating a (meaningless) description of the environment but actively searches for invariants which are coupled to the successful accomplishment of decisions and actions. In the following years, the new paradigm of *active vision* was elaborated by the work of Aloimonos *et al.* (1987), Aloimonos (1993), Ballard (1991), and Blake & Yuille (1992). The main idea was to combine different observations and *a priori* information in such a way that the process achieves a common task. This includes the active control of the sensors. The active computer vision paradigm had several consequences on a system level (Crowley, 1995): (i) continuous operation (the system is always running), (ii) the vision system acts as a "FILTER" for information, (iii) real time (an active vision system must return its results within a fixed delay to be useful), (iv) region of interest (a fixed delay response requires limiting the data).

The ideas of active computer vision boosted two lines of research that are still very much established in the computer vision community. The first is the area of visual attention (Tsotsos, 1989, 1992; Itty & Koch, 2001) that includes a large variety of phenomena and can be characterized as "a mechanism that optimizes the search processes inherent in vision" (Tsotsos & Shubina, 2007). A second line is *robot vision* that leads to tighter integration of robotic and vision control in fields like navigation, obstacle avoidance, scene exploration, grasping, etc. These concepts have been extended towards the understanding of vision as a process (VAP) (Crowley & Christensen, 1995) that includes several sources of contextual information. Rather than reconstructing a scene, these approaches focus the evolution of scenes over time.

Around 2000 the newly emerging terms of 'cognitive vision' and 'cognitive systems' gave these line of research a new twist. Research focuses on the knowledge acquisition process which is fundamentally based on perception. Many of these approaches emphasized the underlying principle of embodiment, i.e. the agent's perception of the environment is shaped by its whole body movement and is coded in terms of the effects on its own body.

Situated computer vision

We are currently witnessing that computer vision is becoming a more and more important cue in Human-machine interaction. Verbal statements relate to the external scene, gestures provide means of non-verbal communication,

actions indicate human intentions, gaze provides hints on human attention. From the standpoint of communication, vision is a rich source of contextual information. But we can also turn the perspective around. Vision takes place in a communicative situation, it provides expectations for visual processing, and it dictates which aspects are relevant. As a consequence, the attention problem becomes a joint attention problem, the scene interpretation problem becomes a problem of joint situation awareness, and the control problem becomes a human-in-the-loop problem.

In the following chapters, computer vision will mostly be treated as a stochastic process. This pays tribute to the inherent uncertainty of relating situational expectations to vision results. Modeling of context plays an important role that should lead to more stable scene interpretations.

From a control perspective, situated computer vision is not one-way. The human communication partner is able to shape the visual capturing and interpretation process. He or she can direct the camera view or provide verbal hints. Therefore, the system needs to provide appropriate feedback and react on an appropriate time scale. Thus, vision becomes an interactive process. It is driven by situation models on the human's as well as on system's side. The more coupled these situation models are, the more successful the visual interpretation process will be.

1.2 Situation models

Situation models have been developed in text comprehension. They are indispensable for understanding the text's content because a pure text-based representation of words, syntax, and semantics typically does not tell the whole story. Many relevant aspects need to be inferred from the text and experimental studies have shown that – while reading – people do form mental representations of the described state of affairs which are termed *mental models* (Johnson-Laird, 1983) or *situation models* (van Dijk & Kintsch, 1983; Zwaan & Radvansky, 1998). The experimentally collected insights on situation models changed or even redefined the role of language from a representational towards an operational character. Instead of directly coding propositional information, it is seen as processing instructions for constructing situation models (Gernsbacher, 1990). Many different theoretical frameworks have been proposed that mostly serve as an explanation of human performance in text comprehension rather than as computational models for

artificial systems:

- *The interactive model of comprehension* (Kintsch & van Dijk, 1978) proceeds in cycles of connecting small sets of propositions extracted in each sentence to existing ones.

- *The construction-integration (CI) model* (Kintsch, 1998) consists of a two-phase process: the construction of mental units and the context-sensitive integration. Therefore, it utilizes a multi-level representation (surface-level, propositional text, situation model).

- *The structure-building-framework* (Gernsbacher, 1990) maps many processes and mechanisms in language comprehension to general cognitive processes. The goal is to build coherent mental structures and the building blocks are memory nodes activated by incoming stimuli. The structure building process is controlled by suppression and enhancement of the activation of memory nodes.

- *The event-indexing model* (Zwaan *et al.*, 1995) represents situation models as a network of event nodes in long-term memory. These events are indexed by five dimensions (space, time, causation, intentionality, protagonists and objects).

- *The immersed experiencer framework* (Zwaan, 2004) includes action and perceptual representations in the situation model and declines amodal propositional representations as a core component.

Situation models are not specific to language. Already van Dijk & Kintsch (1983) argued that situation models "are needed to explain similarities in comprehension performance across modalities." We have a similar understanding of an event regardless if we have read a newspaper article about it or if we have seen it in a news report on television. While early accounts to situation models deduced an amodal propositional representation format from this evidence, Zwaan (2004) emphasizes the cross-modal character of situation models. He assumes that the processing of each word also activates its experienced referent, be it its perceptual or motor representation. He reports various experimental evidences that show a tight coupling between situation models, action representations, and perceptual representations.

In general, situation models serve two main purposes that are shortly discussed in the following subsections. First, they are used in order to integrate

information across sentences in order to build up efficient retrieval structures for the content just gathered. Secondly, they provide strong expectations about the future course of events in order to explain observations or to guide future action selections.

1.2.1 Storage and retrieval structures

If we assume the process of constructing situation models to be driven by different sensor modalities – rather than language only –, it must (to a certain degree) be determined by the way how humans experience their environment. Thus, the continuity of time, space, and perspective needs to play a fundamental role in the internal organization and structure of situation models. Although these continuity assumptions do not necessarily hold for language, experimental results in language comprehension have shown that a violation of these assumptions leads to an increase in reading time and raises the cognitive load in terms of brain activity (as discussed in Zwaan (2004).

According to the event-indexing model (Zwaan *et al.*, 1995), events are the building blocks of integrated situation models, which are indexed by 5 different dimensions: *time, space, causation, motivation,* and *protagonist.* A situation model is, then, stored as a network of event nodes in long-term memory. The event-indexing model puts forward two hypotheses.

- The *memory organization hypothesis* states that the more dimensional indices two events share, the more strongly those events will be associated in memory (Therriault *et al.*, 2006, p. 79).

- According to the *processing load hypothesis* , the fewer indices that are shared between the current event and a previous event, the more difficult it will be for readers to integrate the current event into their situation model (Therriault *et al.*, 2006, p. 79).

Experimental studies presented by Therriault *et al.* (2006) suggest that the *time* and *protagonist* dimensions are more fundamental than space and less affected by the actual task setting. The influence of causation and motivational goals has been explained by Zwaan's immersed experiencer model as being "specific instances of the more general factor of predictability" (Zwaan, 2004, p. 50). The anticipation of an event by a *cause-effect* relation or a *goal/plan* structure leads to an increase of overlap also in the other dimensions.

Most approaches to the construction of situation models assume a multiple step procedure. The event-indexing model starts by extracting event units from language input resulting in a *current model* that is, then, incorporated into an *integrated model*. Besides the guiding of this process by linguistic cues, world knowledge plays a crucial role in it. "Readers make use of their knowledge about experienced situations to construct situation models" (Zwaan & Radvansky, 1998, p. 177). This has also been shown experimentally, by comparing the recall of soccer "experts" with novices in terms of speed and accuracy (Schneider & Körkel, 1989). Thus, the construction of situation models is an interplay between the short-term working memory (STWM) and the long-term working memory (LTWM). Ericsson & Kintsch (1995) assume that the STWM contains retrieval cues to LTWM, which enables people to efficiently access LTWM and thereby expand the STWM. In line with this, Zwaan & Radvansky (1998) propose that the integrated situation model is stored in LTWM while the current model is constructed in STWM.

The immersed experiencer model (Zwaan, 2004) offers a more detailed approach to the model construction process. He distinguishes three components: (1) the *activation* on word level, (2) the *construal* on clause level, and (3) the integration on discourse level.

Activation: Zwaan assumes that the perception of a word activates "functional webs" of neurons that are also activated when the word's referent is experienced. Thus, besides word-related lexical, grammatical, and phonological representations, also referent-related motor, perceptual, and emotional representations are activated. Which parts of a functional web are selected also depends on the semantic and task context in which the word is processed.

Construal: Zwaan builds his framework on *intonation units* which define speech segments that are indicated by prosodic markers like pauses, pitch shifts, or changes in voice quality. The intonation units are processed incrementally and refer to an event defined by a continuous period of time, a spatial region, and a perspective. "The construal is the integration of functional webs in a mental simulation of a specific event" (Zwaan, 2004). Zwaan refers by *construal* to the construction process as well as to the memory representation of such an event. Besides the spatio-temporal components, each construal includes a focal

entity, a relation extracted from a verb or preposition, and optional background entities.

Integration: A situation model consists of several construals coding a network of events. The integration process refers to the transition from one construal to the next one reconstructing an experienced trace of a protagonist. This could be a perceptual transition by zooming, panning, fixating, or a transition that switches from one sensory modality to another, for instance. Hence in each step, the recipient adopts the perspective and attentional frame of the protagonist. These kind of transitions are also modulated by changes in the scene that attract attention.

In summary, text comprehension can be understood as the process of constructing situation models. Information that is linearized into a text needs to be reconstructed along different dimensions that span an appropriate space for embedding and accessing information stored in situations. These comprehend all the entities, relationships, and beliefs that are relevant for an agent operating in its environment. Access structures are organized via the short-term working memory that also includes access patterns indexing activated components of the situation model stored in the long-term working memory. Events seem to be an appropriate building block for constructing situation models which is an incremental process. The ease of integration of newly gathered information depends on a number of factors that suggest certain principles, like cross-modal activation, temporal, spatial, and perspective continuity, overlap, and predictability. These support the hypothesis to understand situation models as a mental simulation of experienced events. Linguistic cues as well as world knowledge are used in order to construct these simulations.

1.2.2 Situation models as a dynamical representation

Situation models are more than a representation of a static state of affairs at a certain point in time. Many of the characteristics discussed before suggest a quite dynamic characteristic. Several experimental results have shown that situation models influence an agent's continuous perception as well as action (Zwaan, 2004):

- Motor areas and visual areas in the brain are activated by words.

- Specific actions are performed faster if they are pre-activated by the situation model.
- Pictures are recognized faster if an object's pose is coherent with a text read before, e.g. an eagle with wings spreaded if text mentions a flying eagle.

These findings suggest that situation models guide the decisions and actions of a recipient. This needs to be a continuous process which is coherent with Zwaan's and Oatley's idea of situation models as a *simulation of experience* (Oatley, 1999).

A first level of dynamics can be identified in the evolution of activation patterns over time. Zwaan (2004) assumes that diffuse patterns of activation induced by words are narrowed down by taking contextual information into account. The context is provided by the mental simulation that constrains possible fits of activated functional webs .

A second level of dynamics is constituted by the prediction capabilities of the situation model. If two sequential events have been observed several times, we tend to predict the second event from the first. Thus, the events are stored as a transition in time rather than two states at separate time steps. The situation model is stored as a prediction dynamics rather than a state of affairs at a discrete time. The dimensions of causation and motivation can be interpreted as more sophisticated prediction models. The concept of causation abstracts from the immediate relation in time and builds transitive chains of events. Motivation introduces a hidden state variable coding a long-term goal that cannot be directly observed but explains a sequence of events. The influence of predictability on the construction of situation models has been experimentally demonstrated by measuring massively reduced response times of recipients during text comprehension (Keefe & McDaniel, 1993).

The dynamic representations that constitute the inferential capabilities of the situation model are an inherent part of it. They are built up during text comprehension but should not be interpreted as static fact knowledge. Rather they should be interpreted as a re-shaping of activation patterns and its underlying dependency structure. Dynamical representations are established on different time-scales and provide efficient building blocks for the construction of new situation models. One example is the establishing of routines that resemble stereotype action sequences. *Routinization* is typically treated as a long term process. Here, a routine is an almost fixed expression that has a particular syntax, meaning, and pragmatics. Conversational

patterns are typical examples like "How do you do?" or "Thank you very much." Pickering & Garrod (2004) also claim that routines are built on-the-fly during dialogue. They cite Aijmer (1996) who estimates "that up to 70% of words in the London-Lund speech corpus occur as part of recurrent word combinations" (Pickering & Garrod, 2004, sec. 5.2). They put forward that the high degree of routinization in dialogue is based on *interactive alignment* mechanisms.

The driving force towards the alignment of intonation, vocabulary, syntactic structures, semantic meaning, and pragmatic use in dialogue are massive priming mechanisms on each level of linguistic representation. As a consequence Pickering & Garrod assume a stimulus-driven rather than an entirely internally generated process of speech production. This contrasts with other models propagating an extensive use of the *theory of mind* (Dennett, 1996) which includes a reasoning about the mental model of the other interlocutor and his/her model of the own mental model and so forth. However, the alignment hypothesis is in line with text comprehension models like the immersed experiencer which includes the activation of functional webs. Pickering & Garrod extend the understanding of text comprehension as the construction of situation models towards the understanding of dialogue as the interactive alignment of situation models. This can only be achieved by an extensive use of contextual cues that do not only influence a semantic processing layer but early perceptual processes, too.

1.3 Context in Human vision

The role of contextual information in the visual processing pathway is a fundamental architectural issue for artificial as well as biological vision systems. On the one hand, findings in biology, psychology, and neuro-science provide hints for building artificial vision systems. On the other hand, artificial systems provide evidence for testing certain architectural hypotheses. As a simulation toolkit, they allow to inspect internal states that are otherwise not accessable. Contextual cues have played a role in human vision research for quite a long time. Early experiments on scene perception and visual search have been conducted e.g. by Palmer (1975), Potter & Levy (1969), or Biederman *et al.* (1982).

Much work has been based on eye-tracking experiments measuring the fixation times and location saccades in human scene exploration. Another

Figure 1.1: The structure of scenes frequently helps in recognizing – in this case – blurred objects (taken from Torralba (2003)).

bunch of work measures the human performance in object detection given a series of photographs of scenes presented in a rapid succession. A third line of research is based on the activity measurement of cortical regions. In the following, I will summarize a few selective results in all three directions.

But first, let us take an introspective perspective based on our own personal judgement. Context plays an important role especially in case of poor viewing quality (large distance, short acquisition times, occlusions, illumination, shadows, peripheral vision leading to poor resolution or contrast). This can easily be experienced by looking at low-pass filtered images. In Fig. 1.1 the isolated object areas cannot be identified because of strong blurring effects. Nevertheless, we can easily identify certain objects in the context of a typical scene layout.

1.3.1 Results from eye-tracking experiments

Eye-tracking experiments gain their interest from the hypothesis that the locus, duration, and sequence of the eye fixations are closely tight to cognitive processes which otherwise are unobservable (Just & Carpenter, 1976). The highest-quality information perceived by the eyes relates to a very small portion of the retina that corresponds to a view angle of 2°. The visual-cognitive system exploits the high resolution power by re-orienting the fixation point in a viewed scene three times a second, on average. This is a partially unconscious process. Eye-tracking studies make the assumption that the current

fixation point in the scene correlates with information currently processed in higher-level scene perception. Henderson & Hollingworth (1999) summerize several results on eye movement studies:

- Fixations are non-random and cluster on both visually and semantically informative regions.
- The first few fixations seem to be controlled by visual features in the scene and global semantic characteristics of the scene.
- Later fixations further analyze local regions controlled by visual and semantic properties of the local regions.
- Studies suggest that no composite image is mentally constructed during fixation saccades. Instead a limited amount of information is carried across saccades that is coded in a relatively abstract format. Thus, the experience of a complete and integrated visual world seems to be an illusion or construction based on the stored abstract conceptual representations.

Thus, global semantic analysis seems to be faster than local semantic analysis and scenes are only perceived, very selectively, partially based on contextual cues.

1.3.2 The object-detection paradigm

Much evidence of the conclusion that scene knowledge interacts with object perception is based on the object-detection paradigm put forward by Biederman *et al.* (1982). Here, participants were briefly presented scenes that either violate semantic relationships with a target object or not (Fig. 1.2). People were able to determine the presence of the target object much more reliable in cases of consistent scene presentations than in violation conditions. As a conclusion, it is assumed that semantic relationships can be accessed very quickly and that *"... stored knowledge about scenes and objects likely to appear in them can be used to facilitate the construction of perceptual descriptions of consistent objects"* (Hollingworth & Henderson, 1998, page 399). This leads to the *perceptual schema model* of scene context effects (Biederman *et al.*, 1982). Alternative explanations are based on a *priming model* of scene context effects as also discussed by Kosslyn (1994). He assumes that

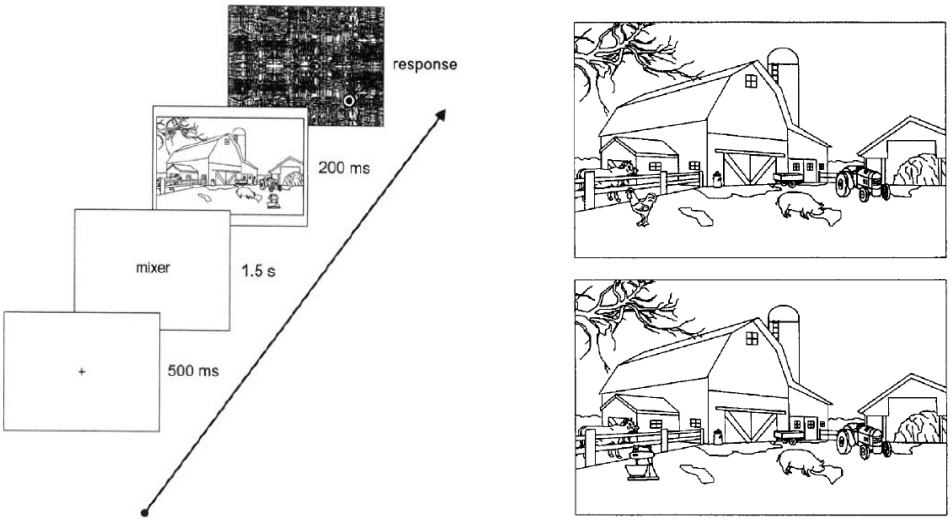


Figure 1.2: Experimental setup for the rapid presentation of photograph series and two typical scenes used in the experiments of Biederman *et al.* (1982). Note that objects like 'chicken' and 'mixer' are combined with consistent and inconsistent scenes. In the experiment, first the label is prompted to the participants and then an either consistent or inconsistent scene is presented containing the object or not. Images have been taken from Hollingworth & Henderson (1998).

object representations in the long-term memory are pre-activated by scene recognition rather than included in schema-like scene representations.

Hollingworth & Henderson criticize conclusions drawn from the object-detection paradigm because pre-processing contextual effects are not isolated from post-identification effects (Hollingworth & Henderson, 1998). They vary the original experiment by Biederman *et al.* (1982) in several dimensions and suggest a functional isolation of object perception from information about which objects are likely to appear in a scene.

1.3.3 Neurophysiological results

Neuroimaging data measures the activity of the brain in different cortical subregions. Certainly, a collection of these is permanently active whether

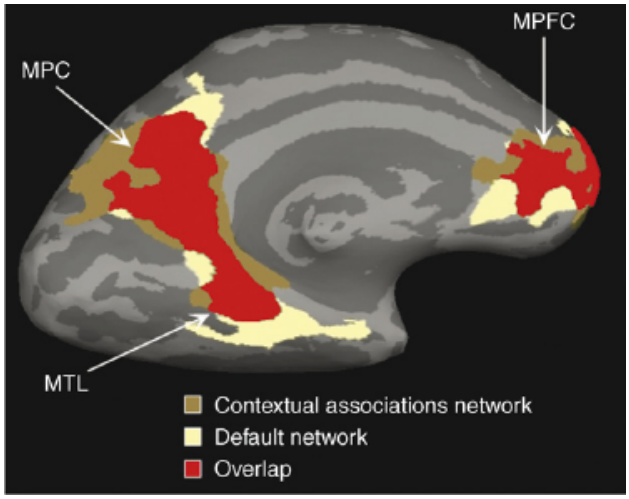


Figure 1.3: Medial view of the typical default network and the typical contextual association network (taken from Bar (2007))

we are performing a task or not. This has been termed the *'default network'* (Bar, 2007). Therefore, the primary method is to substract the signal elicited by one condition from the signal elicited by another condition.

Several studies, like that of Bar (2007), suggest a parallel processing pathway in the human brain that provides fast top-down predictions. It is assumed to be activated by information that are early available in the standard visual pathway (the ventral visual stream), e.g. low spatial frequencies (LSF). The general framework proposed by Bar (2007) is based on three primary components. (1) Associations, which are formed by a lifetime of extracting repeating patterns and statistical regularities from our environment, and storing them in memory; (2) the concept of analogies, in which we seek correspondences between a novel input and existing representations in memory (e.g. 'what does this look like?' rather than 'what is it?'); (3) these analogies activate associated representations that translate into predictions.

In Bar *et al.* (2007) results are reported for contrasting the presentation of objects that are either strongly or weakly associated with regard to a task. Interestingly, there is a striking overlap between the typical contextual association network and the 'default network'. This supports the hypothesis that the brain is continually engaged in the generation of associations-based prediction (Bar, 2007).

The main regions involved in the network of associations-based predic-

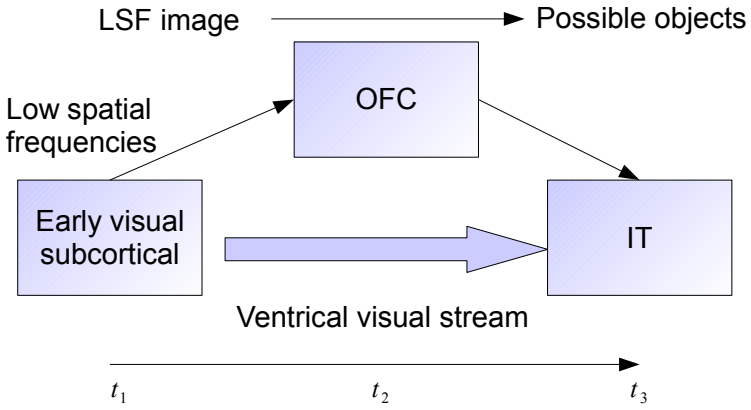


Figure 1.4: A top-down facilitation model. The *'gist'* image activates predictions about candidate objects that are similar to the images in their LSF appearance. The combined contributions of stimulus-specific context ('my kitchen', represented by PHC) and prototypical context (e.g. 'a kitchen') elicits prediction-related representations in the PFC (orbifrontal cortex, OFC, in particular) as well as in a domain-specific cortex such as the fusiform gyrus in the case of object recognition. The ventral pathway runs from V1 to V2, V4 and the inferior temporal cortex (IT) representing the *what*-system.

tions are the medial temporal lobe (MTL), medial parietal cortex (MPC), and medial prefrontal cortex (MPFC) (Fig. 1.3). The MPC represents knowledge about associations related to prototypical context, e.g. where to expect a stove in a standard kitchen, while the parahippocampal cortex (PHC) as part of the MTL represents stimulus-specific context and associations with regard to the specific appearance of *my* kitchen. Their combined contribution elicits prediction-related representations in the prefrontal cortex (PFC) and, in particular, the orbitofrontal cortex (OFC) (Bar, 2007). Fig. 1.4 sketches the interaction assumed between different processing pathways in the brain. The ventral pathway depicted is one of the two theorized systems of visual information processing (Ungerleider & Mishkin, 1982). It provides functions for the analysis of object qualities (the *what*-question) while the *dorsal* pathway comprehends the spatial arrangement (the *where*-question). Moshe Bar and others found an early activation of the OFC before the processing in the ventral pathway reaches the inferior temporal cortex (IT) for higher-level

cognitive processing.

1.4 Summary and conclusion

In this chapter, the term *situated computer vision* has been positioned as relating established computer vision principles to human-machine interaction scenarios. In such scenarios, context plays a fundamental role that influences early visual processing as well as higher level control. Computer vision becomes an interactive process that establishes a continuous feedback loop with a human communication partner. Both interlocutors' perception is influenced by the state of affairs that summarizes the interaction history. In this area, a large body of research deals with situation models and has lately shown that there is a tight coupling between language, perception, and action. Thus, anything that is stored in the situation model from one of the modalities will influence the perception of other modalities. Research on biological vision shows that context is not a second step process but tightly couples with early visual processing. This needs to be taken serious in computer vision techniques, that are explored in the next chapters.

Chapter 2

Perception of Scenes

In this chapter, I will review several computer vision techniques that recognize scene contexts and relate them to object instances occurring in the scene. Dealing with context is a key issue towards *situated* computer vision. First, systems need to deal with complex natural scenes (e.g. living-rooms, kitchens, public places, etc.). Secondly, they should exploit expectations about a scene similar to those of the human interaction partner (e.g. no flying cups or cars). Otherwise it would cause irritations. Thirdly, humans tend to explicitly and implicitly refer to contextual aspects in case of ambiguity (e.g. "the cup is *on the table*", "he is standing *at the right corner* [*of the house on the sidewalk*]"). Rather than reporting a complete enumeration of relevant work in the field, I selectively discuss approaches that focus on different kinds of representations. Scenes can be modeled as holistic entities or configuration of parts, using bag-of-features or relational representations, characterizing them by 2D or 3D features. The following sections give an overview of techniques used in the field. The argumentation will be frequently given along the lines of probability theory and graphical models. In this case, image pixels, patches, or features are represented as random variables. These are observed given an image. Any kind of object or scene parameters are modeled as hidden variables. These need to be inferred by the recognition system.

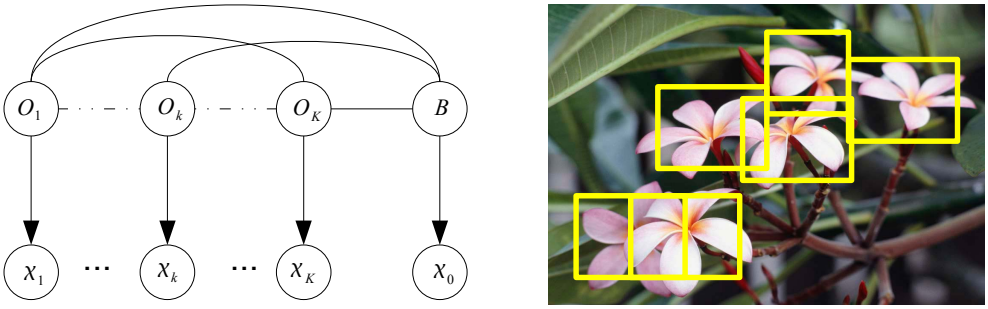


Figure 2.1: Graphical model visualizing the conditional independence assumptions. The random variable sets $\mathcal{X}_k = \{X_{j(l)}\}_{l=1...L_k}$ refer to image portions marked by a yellow bounding box.

2.1 Why context?

Nearly all interpretation tasks of real-world sensor data include a separation of foreground and background because the semantic concepts typically used in interpretations only refer to a certain part of the sensor data. In case of visual data, I will refer to those foreground regions that are bound to a single concept as caused by a unique physical *object*. Given this correspondence between objects and image regions it is very intuitive to make a conditional independence assumption: Given all object related information (like location, pose, size, appearance, etc.) and the mapping between the visible part of the object and the image (coded by region outline(s), bounding box(es), or by camera parameters), the set of image pixels referred to by the object's mapping are statistically independent of the rest of the image (Fig. 2.1).

$$Pr(i(\mathcal{X})|o_{1:K}, b) = Pr(i(\mathcal{X}_1)|o_1) \cdot \ldots \cdot Pr(i(\mathcal{X}_K)|o_K) \cdot Pr(i(\mathcal{X}_0)|b) \quad (2.1)$$

where $i(\mathcal{X})$ is the image, $\mathcal{X}$ is the set of random variables representing all image pixels, $\{o_k\}_{k=1...K}$ code the parameters of all physically separated foreground objects in the image, $\mathcal{X}_k$ is the set of image pixels where object k is visible, and b codes the background which is visible in pixels $\mathcal{X}_0$.

This certainly is an approximation and abstraction from the real plenoptic function. First, the independence assumption does not cover lighting

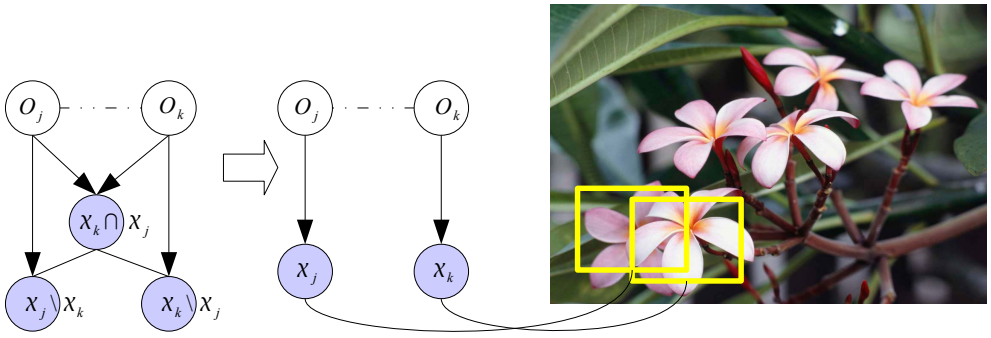


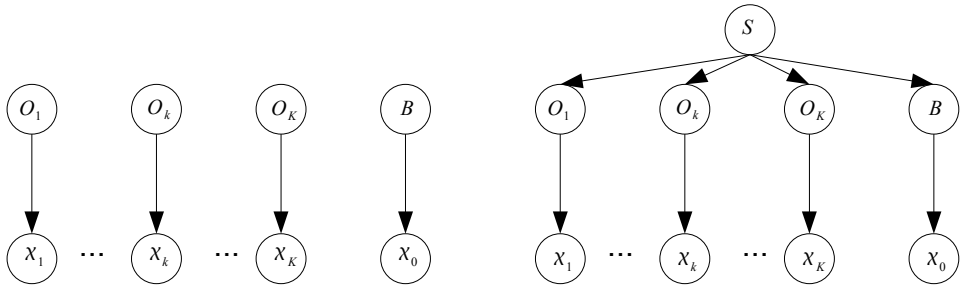
Figure 2.2: Graphical model visualizing the independence assumptions if occlusions are ignored. The evidence of the overlapping bounding boxes is separably considered for both object hypotheses.

conditions that are constant or similar over large portions of the image. Secondly, it ignores lighting effects that project properties of nearby objects or background onto the object's surface caused by transparency, reflections or specularities. Thirdly, it does not model shadows caused by other objects. Partially, the deficiencies can be balanced by applying appropriate image transformations as a pre-processing step or by computing image features that are less variant with regard to lighting effects.

Another shortcoming is the treatment of object occlusions. Occlusions cannot be dealt with in a systematic way because they are either ignored (Fig. 2.2) or need to be coded into the mapping which leads to cumbersome representations. Frequently, image pixels that are related to two overlapping hypotheses are doubly used as independent evidence for both hypotheses.

As a consequence, all variations that are not covered by the abstractions are reflected in the statistics of the object's appearance model $Pr(i(\mathcal{X}_k)|o_k)$. An illustration is given in Fig. 2.1. Although all blooms in the image have a similar color appearance due to some global lighting conditions and a common orientation towards the lighting direction, the model in Eq. 2.1 considers all possible light conditions and all possible poses for each object instance, separately. Furthermore, shadows and occluding objects corrupt the image appearance statistics inside the object's bounding boxes (Fig. 2.2).

In typical computer vision tasks neither the object nor the mapping information is given, rather it is the goal of inference. Thus, the joint distribution



(a) Object independence (b) Conditional object independence

Figure 2.3: Graphical model visualizing the conditional independence assumptions.

over the image and the objects needs to be modeled:

$$Pr(i(\mathcal{X}), o_{1:K}, b) = Pr(i(\mathcal{X})|o_{1:K}, b)Pr(o_{1:K}, b) \tag{2.2}$$

Approaches that ignore context introduce further independence assumptions by treating the objects independently and the background as random clutter:

$$Pr(o_{1:K}, b) = Pr(o_1) \cdot \ldots \cdot Pr(o_K)Pr(b) \tag{2.3}$$

Thus, the object recognition task can be formulated as

$$(\hat{o}_1, \ldots, \hat{o}_{\hat{K}}) = \underset{(o_1,\ldots,o_{\hat{K}})}{\operatorname{argmax}} Pr(i(\mathcal{X}_1)|o_1)Pr(o_1) \ldots Pr(i(\mathcal{X}_{\hat{K}})|o_{\hat{K}})Pr(o_{\hat{K}}) \tag{2.4}$$

where $\hat{K}$ is the number of objects that are reliably detected in the image. Fig. 2.3(a) visualizes the independence assumptions by depicting the corresponding graphical model.

This equation can be solved for each o_k, separately. Or in other words, the image interpretation problem is decomposed into an independent set of object detection problems. Each is formulated as a probabilistic inference task. Independent of the specific object representation, there are two different ways for solving it: *conditioning* and *inverting probabilities*.

- *Template-based and model-based* object recognition approaches use a conditioning strategy by sampling over the object parameters (coded by the O_k random variables) and judging the hypothesis o_k by computing a likelihood $Pr(i(\mathcal{X}_k)|o_k)$ or similar measurement. The sampling strategy might be improved by gradient-ascent methods or other methodologies.

- *Feature-based approaches* try to infer the inverted probability $Pr(o_k|i(\mathcal{X}_k))$. If no previous segmentation is given, some of them even estimate the inverted joint probability $Pr(o_1, \ldots, o_{\hat{K}}|i(\mathcal{X}))$ by accumulating feature evidences for different object hypotheses in parallel (feature-counting approaches). However, these also introduce further independence assumptions on features extracted from $i(\mathcal{X})$.

Isolated object detection or classification methods are successful as long as the likelihood $Pr(i(\mathcal{X}_k)|o_k)$ dominates the max-function over the joint probability term $Pr(i(\mathcal{X}_k), o_1, \ldots, o_K, b)$. For a single object detection problem the two relevant factors can be written as[1]:

$$\hat{o}_k = \underset{o_k}{\operatorname{argmax}}\, Pr(i(\mathcal{X}_k)|o_k) Pr(o_k|b, o_{j=1:K, j\neq k}) \tag{2.5}$$

The equation is dominated by the first factor as long as $i(\mathcal{X}_k)$ includes sufficient information for distinguishing between alternative object hypotheses. It especially becomes critical if the size/resolution of $i(\mathcal{X}_k)$ is too small/low, the configuration space of O_k is very large, the inner-class variations of $Pr(i(\mathcal{X}_k)|o_k)$ are large, or the inter-class separations are fuzzy.

In order to meet these constraints most methods need to be tuned for a specific application domain and some global threshold parameters need to be fixed. If we make these parameters explicit, it results in a somewhat less restricted model shown in Fig. 2.3(b) assuming a conditional independence between the scene objects and the background. By tuning the object recognition parameters for a specific task, we implicitly instantiate the random variable S to a dedicated class of scenes that we are dealing with. In this regard, this model will provide a basis for further discussions on contextual issues in object recognition tasks.

2.1.1 Aspects of contextual modeling

All the independence relations discussed so far abstract from the real image formation process. They ignore interrelationships and dependencies among the random variables. To a certain degree this is a necessary step in order to reduce the parameters of the image formation process, which is a precondition for making many computer vision techniques feasible and applicable.

[1]Those factors that do not depend on o_k have been left out because they do not affect the maximum operation.

However, it also has its limits because systematic variations of an object's appearance, location, and pose is treated as random noise or independent random selections. Context-based vision tries to selectively exploit dependencies ignored before. In a frequently cited article, Strat & Fischler (1991) distinguish different kinds of contexts that partially tackle the deficiencies discussed above:

Global contexts: Many aspects like color selection, the scale, fractal dimension and smoothness of image structures, expected object categories, etc. are determined by global scene information or classes, like daytime, landscape, city, indoor, sunset, etc. This corresponds to the variable S introduced in Fig. 2.3(b). However, the graphical model depicted is a simplified version because the global context could also influence the appearance models of the different object instances. The same argument also counts for movements, activities, and actions observed in a scene context, e.g. we expect typical ball movements, player activities, and kicking actions in the context of a soccer game.

Location: The location and pose of scene objects is typically quite constraint. On the one hand, the global scene class biases typical spatial configurations of objects, e.g. in a landscape picture the sky is on top and grass or woods are at the bottom, in a city street scene houses are on the left and right. Besides the global spatial layout there are many dependencies between the objects, e.g. one object supports another, the keyboard is in front of the monitor, cars can be found on streets, chairs are located at a table. In dynamic scenes, objects are moving on smooth trajectories.

Appearance: Although the lighting can change in a scene, two objects nearby frequently share the same lighting conditions. Thus, both appearance models are coupled by a common source. This can be exploited, for example, by using face detectors for adapting a skin-color model. The color model can, then in turn, be used to detect other parts of the body like hands. Many object tracking algorithms make the assumption that the lighting is roughly constant over time or changes only smoothly from frame to frame. In this case, they adapt the appearance model based on the trajectory data acquired so far.

Functionality: Functional relationships between objects frequently determine spatial constraints between object locations and relative poses as

well as object class compatibilities. They superimpose a role on each interactor. A bridge that *supports* a car would be a typical example. While the appearance of bridges has a very large variance, the common abstract geometric description (a horizontal line or bow) is not discriminative enough with regard to other object classes or clutter. In this case, the contextual term of Eq. 2.5 dominates the maximum function over the joint probability. A bridge has support on both sides, bypasses a gap, supports other objects like cars, and connects two ends of a street or path. Besides these kind of static roles and relations, objects also have dynamic roles in the context of an action. For example, a cup is used for drinking and has the passive role of a bin while pouring.

Tightly related to functionality, ***task contexts*** are frequently modeled for the interpretation of (human) actions. Compared to functional contexts, tasks cover a longer period of time with possibly changing roles of objects. Tasks structure the temporal evolution of the state of affairs observed and, therefore, provide strong expectations about objects, events, and actions involved.

Finally, communicative cues represent a rich source of contextual information for the interpretation of sensory input. In related work, this has frequently been referred to as ***linguistic context*** (Srihari, 1995). It is typically provided by adjunct text, speech, writing, or even communicative gestures like pointing. When considering this kind of context, the major challenge consists in establishing correspondences between both types of descriptions. This includes the vocabulary as well as structures because in both cases we cannot assume a one-to-one mapping.

2.1.2 Contextual modeling for scene understanding

A further aspect of modeling context is that we might be interested in the contextual information as such, i.e. the relational content that is not directly extractable from image features but can be coded in contextual models. We do not perceive an image or image sequence as a random collection of (arbitrarily moving) objects but associate causal, structural, or spatial relationships, assign roles, and categorize the whole setting. Termed differently, we understand what is *happening* even if only a single still image is presented.

Categorizing scenes

A simplified case of scene understanding is ***scene categorization***. In this case, the system needs to provide a single label (out of L labels) for a scene observed rather than providing a structural description of it. Typically, such approaches have been realized for distinguishing indoor/outdoor, city/landscape, street/house-block/side-walk or photo/sketch categories. In a frequently used approach, a common reason or source s is assumed that explains objects and background. The corresponding random variable S has a previously known number of states coding different scene categories. The model is typically learnt in a supervised manner. If the common reasons are exclusive, this implies a conditional independence between objects and background as already discussed before (cf. Fig. 2.3(b)):

$$Pr(s|i(\mathcal{X}), o_{1:K}, b) = \frac{Pr(i(\mathcal{X}), o_{1:K}, b, s)}{Pr(i(\mathcal{X}), o_{1:K}, b)} \tag{2.6}$$

$$= \frac{Pr(i(\mathcal{X})|o_{1:K}, b)Pr(o_1|s)\ldots Pr(o_K|s)Pr(b|s)Pr(s)}{Pr(i(\mathcal{X}), o_{1:K}, b)} \tag{2.7}$$

The conditional independence assumption implies that the different events $\{o_k\}_{k=1:K}$ are interchangeable given s, i.e. all structural dependencies are already coded in the common source s. This is also known as the *bag-of-words* or *bag-of-features* approach. This is a severe restriction and prevents the use of the model for a relational scene analysis.

Unsupervised categorizations have also been explored in image databases where it is used for navigation purposes in content-based image retrieval (Flickner *et al.*, 1995; Käster *et al.*, 2004; Vasconcelos, 2007). Here, the ultimate goal is to judge the semantic similarity by image distance metrics.

Related to this area, ***place recognition*** deals with identifying an individual location visited before. This information is typically used for topological navigation tasks in robotics (Kosecká *et al.*, 2003; Schubert *et al.*, 2007) and has also been applied in augmented reality settings (Torralba *et al.*, 2003).

Understanding scenes

In case of ***scene understanding***, we aim at extracting more complex descriptions of images like, 'the woman feeds the dog' or 'the car crosses the street'. Taking a more descriptive standpoint, many researchers formulate

Linguistic description:

- The woman is feeding the dog.

Generating observable predictions on different levels of abstraction:

- dog bowl will touch ground; hand will leave dog bowl; dog will touch food; ...
- woman will put dog bowl; dog will eat food; ...
- dog's behavior will change from hungry to comfortable;

Figure 2.4: Scene understanding as automating linguistic description versus prediction engine. Note that the predictions do not need to be verbalized for understanding the scene, rather that scene understanding provides expectations of future events (that have been verbalized here for the reader).

the ultimate goal of image understanding as "automating linguistic descriptions of scenes" (Buxton & Neumann, 1996; Nagel, 1988). A slightly different standpoint is provided, if we recall the spirit of research in situation models summarized in chapter 1.2. Here, the emphasis is on a prediction engine rather than a description engine. The ultimate goal would be to provide expectations of future events on different time scales and levels of abstraction. This includes predictions from a single still image (Fig. 2.4). The verbalization of the qualitative predictions would be a second stage process that is nevertheless an important benchmark of such a system.

As the majority of the work conducted in this area relates to the descriptive standpoint, symbolic reasoning techniques dominate this area because these are tightly related to linguistic descriptions. By introducing an intermediate level of primitive symbols that either directly – or via a geometrical description – refer to the signal domain, different kinds of contextual dependencies can be modeled independently of each other in separate stages (Fig. 2.5). However, many approaches start modeling contextual dependencies at the level of symbolic descriptions of the image and leave the low-level interpretation process with an independent object – or geometric primitive – recognition approach. A main challenge that needs to be faced by all of these

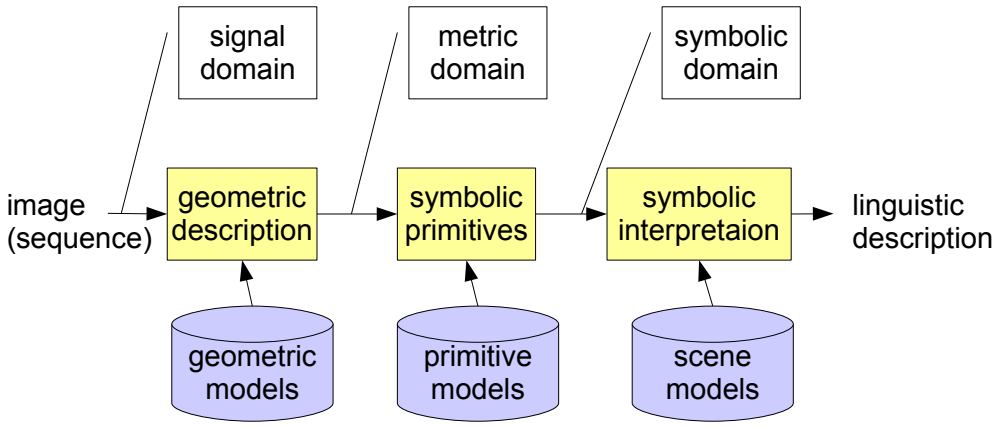


Figure 2.5: Intermediate representations for high-level scene interpretations. Symbolic scene models completely abstract from metric and signal-based context dependencies.

systems is the metric definition of symbolic predicates. Besides uncertain intermediate results, the processing formalism also needs to deal with vague meaning and context-dependent reference frames. In the following, we will not deal with full-blown image understanding approaches but focus more on the interaction between the low-level interpretation process and higher-level contextual information.

2.1.3 Contextual modeling for system control

So far, we concentrated on contextual domain modeling. The probabilistic approach allows us to infer the system's belief in an un-observed domain state given the complete set of observations. This is an intensional perspective and there are good theoretical reasons for choosing such an approach.[2] However, in many cases probability theory is not sufficient. It does not include the concept to actively explore the observations and to take decisions which observational feature should be computed first.

In terms of efficient sequences of system operations, it is sometimes more

[2] Intensional systems provide much more consistent and well founded techniques for dealing with uncertainty. Extensional system have some semantic deficiencies in combining uncertainties as discussed by Pearl (1988).

convenient to think about a problem in terms of the reasoning process. Especially, when the system has to deal with large hypotheses spaces that need to be explored by expensive processing operations – which is the case for many image processing tasks or for robot movements –, heuristic strategies for operation control can be formulated quite fast. A simple representative are rule-based systems that automatically trigger further processing steps based on contextual cues and previous processing results. The early VISIONS (Hanson & Riseman, 1987), CONDOR (Strat & Fischler, 1991), and SPAM (McKeown *et al.*, 1985) systems are typical examples of this approach. The Schema system (Draper *et al.*, 1989) extends it towards a functional decomposition of the system behavior into units combining knowledge representation with procedural knowledge how to apply it.

Active exploration strategies also can be formulated in intensional models. In this case, probability theory needs to be combined with decision theory. In graphical models a special node type is introduced that models possible actions which lead to a new domain state. Then, an action selection is judged by a payoff or utility measurement that is defined on a future state of the model. An example has been put forward by Rimey & Brown (1992). They call it *hypothesis-driven sufficient vision*. The search space is drastically reduced by processing only small portions of an image and interpreting only sufficient detail using knowledge in early processing steps, actively controlling the sensor, and solving specific visual tasks. As a consequence, not all possible interpretations are considered and task specific operations are applied to dedicated image portions. Here, context is applied to control an active exploration process of a scene. Decisions are taken in a separate model of the reasoning process, but it is partially based on evidence gathered in intensional sub-systems which are based on domain models. A more coherent theoretical framework for combining probability and decision theory is given by influence diagrams and Markov-Decision Processes (MDPs). These have been applied in many similar cases (Jensen & Nielsen, 2007).

Both types of extensional and intensional control systems are more extensively discussed in Chap. 5. For the aim of scene interpretation there are three different alternatives for applying contextual knowledge (Fig. 2.6):

Selection: Contextual knowledge is used to pre-select specific image regions or image operations that are applied in the following.

Classification: Contextual cues are directly included in the feature description and used for classification in a single step procedure.

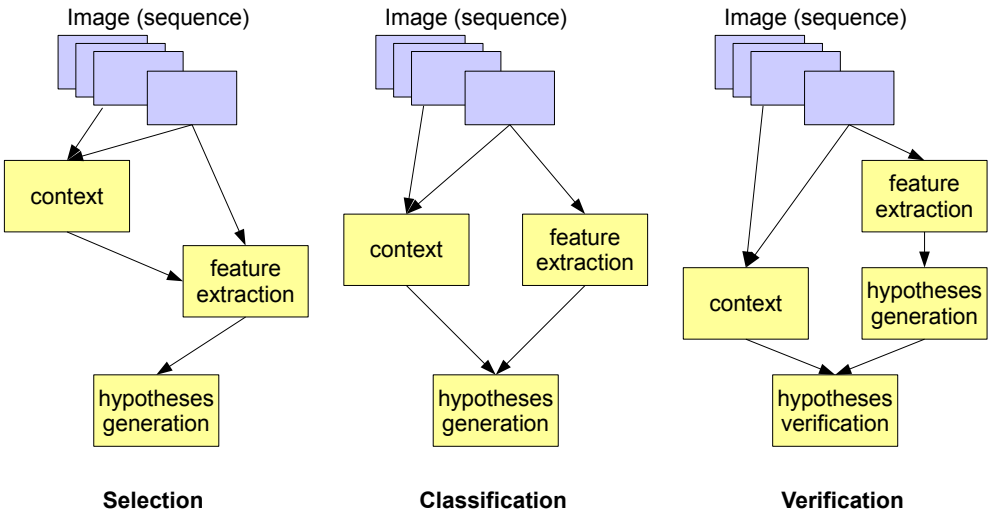


Figure 2.6: Three different control strategies for applying contextual knowledge.

Verification: Contextual knowledge is used in order to judge hypotheses generated by a separate interpretation step.

Each strategy has its own benefits and drawbacks. The first one provides a hard decision which portion of the input data is ignored or which analysis operations are abandoned. Besides the reduction of processing costs, this strategy prevents processing operations from dealing with invalid input data, i.e. sensor data that the operation was not designed for. Simultaneously, it takes the risk of missing valid input data. The third strategy, includes a hard decision on the other side, i.e. hypothesis detection. Only those hypotheses that pass the detection threshold are passed to the verification step. However, the detection threshold is very difficult to balance with regard to unfiltered input data including valid as well as invalid cases. Furthermore, the hypothesis generation is an abstraction step. Thus, the confidence of the hypothesis is typically summarized by a unique, context-independent number (binary or scale), which limits the information basis and influence of the contextual reasoning part. The second strategy is the most flexible but also most expensive approach. All information is available on a feature level in order to infer the most probable explanation. However, one needs to structure the decision problem in order to make it tractable.

2.2 Recognizing global scene contexts

From a technical viewpoint, scene classification is tightly related to object classification in that a single label is provided for an input pattern. However, in designing appropriate models and features one needs to consider that scene labels are provided on a different level of abstraction than object labels. Scenes do not necessarily have a physical connectedness and boundedness that gives its appearance an "all in one piece" look. They are no manipulable items but rather a place where the observer can move. Hollingworth & Henderson (1998, page 244) define scenes from the perspective of human scene perception:

Def. scene (1): *"...semantically coherent (and often namable) view of a real-world environment comprising background elements and multiple discrete objects arranged in a spatially licensed manner. Background elements are taken to be larger-scale, immovable surfaces and structures, such as ground, walls, floors, and mountains, whereas objects are smaller-scale discrete entities that are manipulable (e.g. can be moved) within the scene."*

Oliva & Torralba (2001) contrast scenes with regard to *objects* and *textures* based on the absolute distance between the observer and his/her fixated zone:

Def. scene (2): *"An image represents an 'object' when the view subtends 1 to 2 meters around the observer (hand distance), a 'view on a scene' begins when there is actually a larger space between the observer and the fixated point, usually above 5 meters (scene is characterized as a place in which we can move."*

Both definitions are neither precise nor clearly discriminative. The same image of a table could refer to the *object* 'table' or to the *scene* on the table. This is more or less a question of the spatial scale one is focusing on. The semantic meaning of a scene is determined by its components and relations between the components. Together, both constitute the actions and tasks expected to happen and provide the basic entities for verbal descriptions.

Object-centered vs. holistic schemes

Because of the central role of objects and relations, many approaches choose this level for recognition purposes. I will refer to these kinds of techniques

as *object-centered*. The strength as well as the weakness of this approach is that they require a previous object recognition step. On the positive side it enables a generic and compositional approach to modeling contexts, on the negative side it is an error-prone strategy that especially suffers from segmentation difficulties and late integration of context.

A contrasting approach is based on the *gist* of a scene. The notion of the gist has already been introduced by Friedman (1979) who conducted eye fixation experiments suggesting the *frame theory* as a basic framework for knowledge representation and use. He concludes that *"... subjects might be able to identify expected objects by using automatized encoding procedures that operate on global physical features"* (Friedman, 1979). The gist spans all levels of visual information from low-level features to intermediate and high-level information. As summarized by Oliva (2005, page 251):

> "*[The] perceptual gist refers to the structural representation of a scene which is built during perception. Conceptual gist includes the semantic information that is inferred while viewing a scene or shortly after the scene has disappeared from the view. Conceptual gist is enriched and modified as the perceptual information bubbles up from early stages of visual processing.*"

Being directly grounded in global features the gist bypasses object recognition and activates scene-related knowledge. Oliva & Torralba (2001) term such a low-level feature representation of environmental scenes the *Spatial Envelope*. It is made by a composite set of boundaries, like walls, sections, ground, elevation, slant. Thereby, it represents the relationship between the outlines of the surfaces and their properties including the inner textured pattern generated by windows, trees, cars, people, etc.

Specific vs. generic contexts

The technique chosen in a particular case also depends on the level of abstraction focused on. In categorization research, three different levels are typically distinguished (Rosch & Mervis, 1975):

- *subordinate level:* More specialized categories, like 'car scenes' or 'people in a street'. Here local structures are typically more relevant than global structures of a scene.

- *basic level:* This is the most common access level in the categorization hierarchy, e.g. 'street', 'forest', 'mountain', 'office'. Members of such categories typically have similar shapes or functions.

- *superordinate level:* These are more broad categories that group basic level categories by some common properties, like 'indoor'/'outdoor', 'urban space'/'natural landscape'.

In the following, we will review some techniques that analyze the local and global structures of scenes.

2.2.1 Holistic scene classification

In order to identify meaningful dimensions of the scene structure, Oliva & Torralba (2001) asked 17 observers to split 81 pictures into groups (similar global aspects/structure/elements, not objects or semantic groups). Afterwards the observers were asked about their criteria used for grouping. Oliva & Torralba report 8 different dimensions (naturalness, openness, perspective, size, diagonal plane, depth, symmetry, contrast) the test persons came up with. From these they propose 5 spatial envelope properties:

1. *degree of naturalness:* dominating straight horizontal and vertical lines;

2. *degree of openness:* sense of enclosure, the number of boundary elements increases;

3. *degree of roughness:* refers to the size of its major components;

4. *degree of expansion:* man-made structures are composed of vertical and horizontal structures. The convergence of parallel lines gives the perception of the depth gradient of space;

5. *degree of ruggedness:* deviation of the ground with respect to horizon.

These properties provide a generic means of scene categorization without fixing the specific category labels beforehand.

Second order statistics (energy spectrum)

One possibility to relate these properties to image features is to look at the local and global spectrum of spatial frequencies in an image. Oliva and Torralba propose a spatial distribution of spectral information by means of the windowed Fourier transform (WFT at 8×8 spatial locations with large overlapping neighborhoods, with a diameter of 64 pixels each). Their analysis concentrates on the second order statistics given by the energy spectrum $A(f_x, f_y)^2$ (global spectrum) and $A(x, y, f_x, f_y)^2$ (local spectrum), respectively. These can be computed from the (windowed) discrete Fourier transforms

$$I(f_x, f_y) = \sum_{x=0}^{L_x-1} \sum_{y=0}^{L_y-1} i(x, y)h(x, y)e^{-j2\pi(f_x x/L_x + f_y y/L_y)} \tag{2.8}$$

$$I_{x,y}(f_x, f_y) = \sum_{x'=0}^{L_x-1} \sum_{y'=0}^{L_y-1} i(x', y')h_r(x' - x, y' - y)e^{-j2\pi(f_x x'/L_x + f_y y'/L_y)} \tag{2.9}$$

$$A(x, y, f_x, f_y)^2 = |I_{x,y}(f_x, f_y)|^2; \quad A(f_x, f_y)^2 = |I(f_x, f_y)|^2 \tag{2.10}$$

where $h(x, y)$ and $h_r(x', y')$ are hamming windows with a circular support that is bounded by parameter r in the local case. In the global case it is used to reduce boundary effects. $L_x \times L_y$ denotes the size of the image.

Oliva and Torralba study the unlocalized energy spectrum in more detail for different scene categories and show that the frequency spectrum provides discriminative characteristics (Fig. 2.7). Therefore, they approximate the spectral signatures by a function:

$$E[A(f, \theta)^2|S] \approx \Gamma_s(\theta)/f^{-\alpha_s(\theta)} \tag{2.11}$$

where $E[A(f, \theta)^2|S]$ is the expected value of the energy spectrum for a set of pictures belonging to the category S. Spatial frequencies are represented in polar coordinates (f, θ). The functions $\Gamma(\theta)$ and $\alpha(\theta)$ are obtained by linear fitting of the averaged energy spectrum on logarithmic units for each orientation θ.

The function $\Gamma(\theta)$ reveals the dominant orientations of a scene category. The function $\alpha(\theta)$ represents the slope of the decreasing energy values, from low

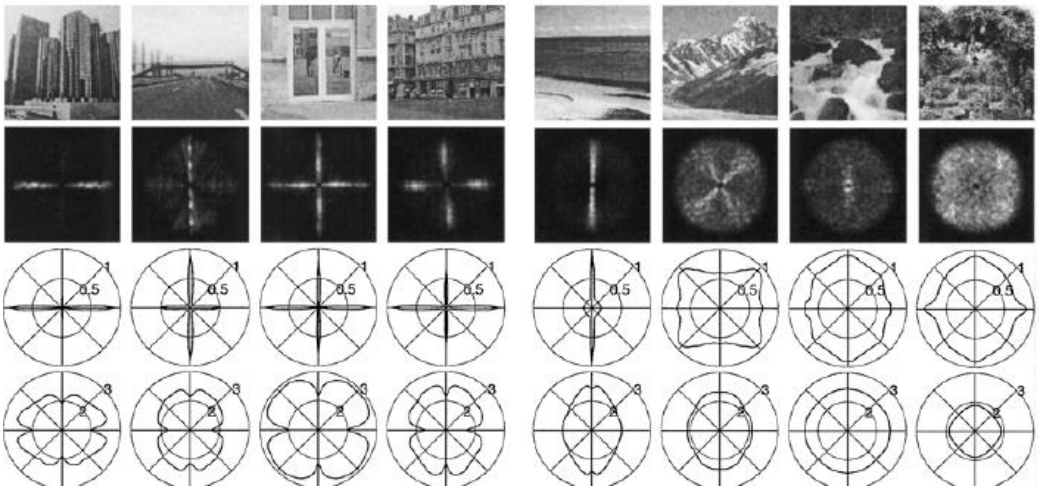


Figure 2.7: Examples of scenes from different categories (tall buildings, highway, urban close-up views, city center, coast, mountain, natural close up views, forest) and their respective energy spectrum (multiplied by f^2 to enhance visibility). Top-row: function $\Gamma_s(\theta)$, bottom-row: function $\alpha_s(\theta)$, (Eq. 2.11) both in a polar diagram (Image taken from Oliva & Torralba (2001)).

to high spatial frequencies (measures the complexity of a scene or fractal dimension / surface roughness, $\alpha \approx 1$ means textured and detailed objects, $\alpha \approx 3$ means large objects and smooth edges, see Fig. 2.7).

In order to map the properties of the spatial envelope to features computed from the energy spectra, first, a PCA is applied on the global and local spectra in order to reduce their dimensionality to N_G (number of global basis functions ψ_i) and N_L (number of local basis functions Ψ_i) components, respectively:

$$A(f_x, f_y)^2 \approx \sum_{i=1}^{N_G} v_i \psi_i(f_x, f_y) \tag{2.12}$$

$$A(x, y, f_x, f_y)^2 \approx \sum_{i=1}^{N_L} w_i \Psi_i(x, y, f_x, f_y) \tag{2.13}$$

The global coefficients $\mathbf{v} = \{v_i\}_{i=1..N_G}$ provide unlocalized structural information while the local coefficients $\mathbf{w} = \{w_i\}_{i=1..N_L}$ capture structural information with a description of the spatial arrangement. Spatial envelope

properties represent the scene in a very low dimensional space in which each dimension depicts a meaningful property of the space of the scene. In order to learn a discriminant spectral template (DST) and a windowed discriminant spectral template (WDST) Oliva and Torralba sort 500 randomly selected images along the axes of a spatial envelope property providing values $\{s_t\}_{t=1..500}$ for each $\{\mathbf{v}_t\}_{t=1..500}$ and $\{\mathbf{w}_t\}_{t=1..500}$. Then they solve a linear regression problem $\hat{s}_t = \mathbf{v}_t^T\mathbf{d} + d_0$ where d_0 is a constant:

$$\mathbf{d}_1 = (\mathbf{V}_1\mathbf{V}_1^T)^{-1}\mathbf{V}_1\mathbf{s}$$

where the vector $\mathbf{d}_1 = [\mathbf{d}; d_0]$ contains the DST/WDST parameters and the constant term d_0; $\mathbf{V}_1$ is a matrix with columns $[\mathbf{v}_t; 1]$; $\mathbf{s}$ is a vector with components s_t for each image t of the training set.

The parameter vector $\mathbf{d}_1$ determines the weighting of features relevant for each spatial envelope property. In case of a binary classification variable $s_t = -1$ or $s_t = 1$ is assigned.

Instead of computing the attributes in the frequency domain, Oliva and Torralba transfer the $DST(f_x, f_y)$ function into the spatial domain by dividing it into two positive functions $DST_+(f_x, f_y)$ and $DST_-(f_x, f_y)$ and computing the impulse responses $h_+(x, y)$ and $h_-(x, y)$ with transfer functions $|H_+(f_x, f_y)|^2 = DST_+(f_x, f_y)$ and $|H_-(f_x, f_y)|^2 = DST_-(f_x, f_y)$.

$$a(x, y) = [i(x, y) * h_+(x, y)]^2 - [i(x, y) * h_-(x, y)]^2.$$

$a(x, y)$ is called the *opponent energy image* and encodes how each spatial location contributes to the attribute s: $\hat{s} = \sum_{x,y} a(x, y)$.

Using the spatial envelope attributes for scene classification, Oliva & Torralba achieve above 90% correctly classified scenes on 'man-made' vs. 'natural landscape'. In this case, the global information seems to be already sufficient. Local information improves the results for the ordering dimensions *ruggedness*, *roughness*, and *openness* which are roughly on the same agreement level[3] as different human subjects.

This approach is similar to using oriented band pass filters (such as Garbor filters or steerable pyramids). In Torralba (2003) a set of complex Garbor

[3]Orderings were compared by measuring the Spearman rank correlation. The Agreement value corresponds to the mean Spearman rank correlation between orderings given by different subjects. Agreement measures of 0.88/0.79/0.86 are reported for openness/ruggedness/roughness with a value of 1.00 denoting a perfect agreement.

(a) Dictionary of 13 filters

(b) Dictionary of 30 spatial templates

Figure 2.8: Dictionaries of filters and spatial templates: Filter 1 is a delta function, 2-7 are 3×3 Gaussian derivatives with different orientations, 8 is a 3×3 Laplacian, 9 is a 5×5 corner detector, 10-13 are long vertical and horizontal edge detectors of size $3 \times 5, 3 \times 7, 5 \times 3$, and 7×3, respectively. Template 1 is the whole patch, 2-7 are half-size sub-patches, 8-30 are one-third sub-patches.

filters tuned to different spatial frequencies were used. The filters were organized in 4 frequencies and 6 orientations. In an alternative approach, Torralba *et al.* (2003) apply a steerable pyramid with 6 orientations and 4 scales. Good results have been achieved by using the first 80 PCA components. They try to classify 63 different places and use an HMM to solve the localization problem. The appearance of each place is modelled by a set of K views (a mixture of K spherical Gaussians with $K = 100$ and $\sigma_p = 0.05$ found by cross-validation). They achieve a precision for 63 different places above 90% for a recall up to 50%. They also train their system on categories (17 instead of 63). Tests on unfamiliar indoor environments (e.g. kitchen, conference room, elevator, corridor, office, lobby) give a significantly worse recognition performance of above 55% precision at 50% recall. Without HMM it even breaks down below 30% precision at 50% recall. Nevertheless, the filter bank works significantly better than PCA performed directly on grey scale images or color images. For categories, color is shown to perform worst and is around the random baseline.

Boosted filter dictionaries

Murphy *et al.* (2003) use a slightly different approach in order to generate appropriate features for scene classification . They define a dictionary of 13 filters (Fig.2.8(a)), that encode different kinds of gradient information and are applied on two different image scales. The filter responses are integrated

using different spatial templates (Fig. 2.8(b)). Instead of a regular grid as used before these define significantly overlapping image regions of different aspect ratios. The response defined for a feature $f(k)$ is the variance ($\gamma_k = 2$) or the kurtosis ($\gamma_k = 4$) of the integrated image region:

$$f(k) = \sum_x w_k(x)|I(x) * g_k(x)|^{\gamma_k} \tag{2.14}$$

where $w_k(x)$ is the spatial template, $g_k(x)$ is the filter, and γ_k is the variance/kurtosis selected for feature k.

These can efficiently be computed by using integral images. The resulting gist has a size of $13 \times 30 \times 2 \times 2 = 1560$ dimensions.

A scene classifier is trained using boosting (in this case GentleBoost (Friedman *et al.*, 2000)). The weak classifier is defined by optimizing a threshold θ for a single component v_f of the feature vector:

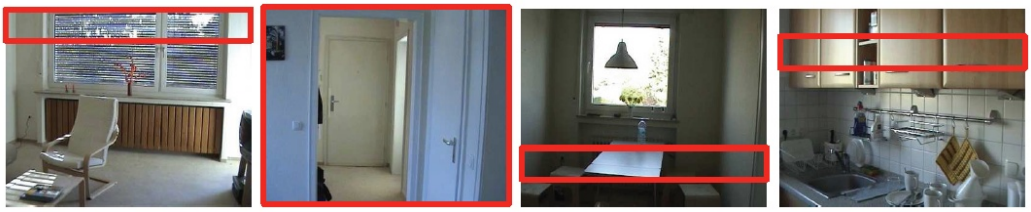
$$h(v) = a[v_f > \theta] + b, \text{ where } [v_f > \theta] = 1 \text{ iff } v_f \text{ is above } \theta, \text{ otherwise } 0. \tag{2.15}$$

The overall classifier is given by a weighted sum of the confidence values of the weak classifiers

$$\alpha(v) = \sum_t \alpha_t h_t(v) \tag{2.16}$$

In Murphy *et al.* (2003), they learn a one-vs-all binary classifier for three different scene types (office, corridor, street) achieving a precision around 95% for 50% recall. Besides, applying the approach to whole images for scene classification they also use it on image patches for the purpose of object classification. This results in a nice homogeneous framework that characterizes the scene on a global as well as local scale.

A similar method has also been applied for a qualitative localization in the context of mobile robotics. Schubert *et al.* (2007) distinguish four different room types in the apartment (living, hallway, dinner, kitchen, see Fig. 2.9). They use 12 different filter responses (11 edge, 1 corner) and the image intensity itself. Integration is performed on 46 differently sized and overlapping image regions (resulting in $13 \times 46 \times 2 = 1196$ feature dimensions). Discrete AdaBoost (Friedman *et al.*, 2000) is used for training a classifier that achieves between 75% and 90% correct classification rate. Because of the relevance in human-robot interaction scenarios, Schubert et al. also examine the effects of a human standing in the view of a camera. In this case, the



(a) Exemplary view of the four room types

(b) Typical filter responses for each room type

Figure 2.9: Features used for room type classification. The red boxes denote the integration region of the most discriminative features for each room type: 5px horizontal edges (living room), absence of 7px horizontal edges (hallway), low light intensity (dinner room), high light intensity (kitchen).

unexpected occlusion introduced by the human body leads to a significant drop in recognition rate. This is an inherent problem of the method that integrates features over large image regions. Schubert *et al.* (2007) propose a rejection criterion for these cases based on a linear classifier: $c = w^T x$, with rejection in case of $c < 0$. The parameters w are learnt on a labelled test set using the perceptron rule. x includes several heuristically chosen features:

- the distances between data feature and threshold of the 4 best weak classifiers $h_t(v)$,
- the sum of these distances,
- the confidence of the strong classifier $\alpha(v)$,
- the difference of the strong classifier to those of the other classes.

The rejection criterion can stabilize scene classification results especially when we evaluate these over several frames in time.

3D features

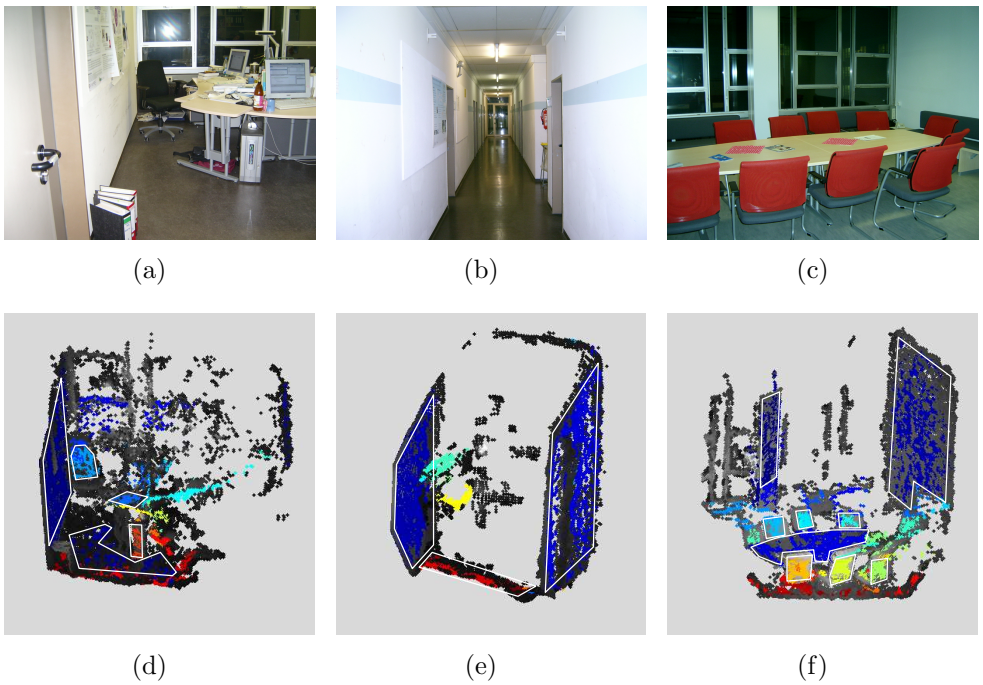
In terms of scene categorization, the power of holistic approaches using only 2D image information is limited. While it works well on the superordinate level ('outdoor'/'indoor', 'urban'/'natural landscape'), especially indoor environments unseen before are problematic for a robust classification of room types (e.g. 'office'/'meeting room'/'hallway'/'living room', etc.):

- Typical furniture like tables, chairs, or shelfs re-occur in different room types,
- Furniture and decorations in the same type of room might have changing colors, textures, and styles (different fabrics, wallpapers, materials) that also cause different appearances in brightness as well as the frequency spectrum,
- The appearance of rooms is highly dependent on view directions. This is much less the case for e.g. landscapes because of a larger viewing distance.

Instead of relying on the appearance of surfaces, room types are much more determined by the spatial layout, that is defined by the surfaces itself. At the same time, there is a large in-class variability in the detailed positioning of furniture in a room. Swadzba & Wachsmuth (2008) propose a holistic approach for classifying room types based on 3D spatial information. They use a Time-of-Flight (ToF) sensor that consists of 176×144 CMOS pixel sensors that directly measure the distance between the optical center of the camera and the real 3D point. Besides the distance value, each pixel sensor provides the amplitude of the reflected near-infrared signal which indicates the amount of light reflected.

The data is pre-processed for noise reduction and filtering of valid points. Then, planar patches are extracted using a combination of seeded region growing and RANSAC. Finally, close-by in-plane patches are merged together forming larger patches (Fig. 2.10). Features are defined from statistical properties of planar patch configurations. Histograms are computed for

- FV1: relative numbers of points per patch (bins are defined on an exponential scale: $b_0 = 0, b_{6+i} = e^i | i \in \{-5, -4.5, -4, \ldots, 0\}$),



(a) (b) (c)

(d) (e) (f)

Figure 2.10: Photos and resulting planes in the corresponding 3D point clouds of (a) an office, (b) a hall, and (c) a meeting room. Each plane is indicated by its own color. Significantly large planes are also highlighted by polygons for gray-colored prints.

- FV2a: angles between patches (5 equally sized bins) – later, the histogram is reduced to the median (FV2b),
- FV3: angles between close-by patches (5 equally sized bins),
- FV4: ratios between sizes of patches (5 equally sized bins).

The averaged histograms for three different room types (office, meeting room, hallway) are shown in Fig. 2.11. Swadzba & Wachsmuth (2008) report two different experiments with different classifiers: nearest neighbor (NN), support vector machine (SVM), Gaussian mixture model (GMM). The first one focuses on the classification of different room types ('office'/'meeting room'/'hallway'). Here, FV1 is the most discriminative feature vector for the classification of an already seen rooms (around 90%), while FV3 is most discriminative for environments unseen before (up to 70%). The combination of FV1,FV2b,FV3,FV4 achieved a correct classification rate up to 99% for

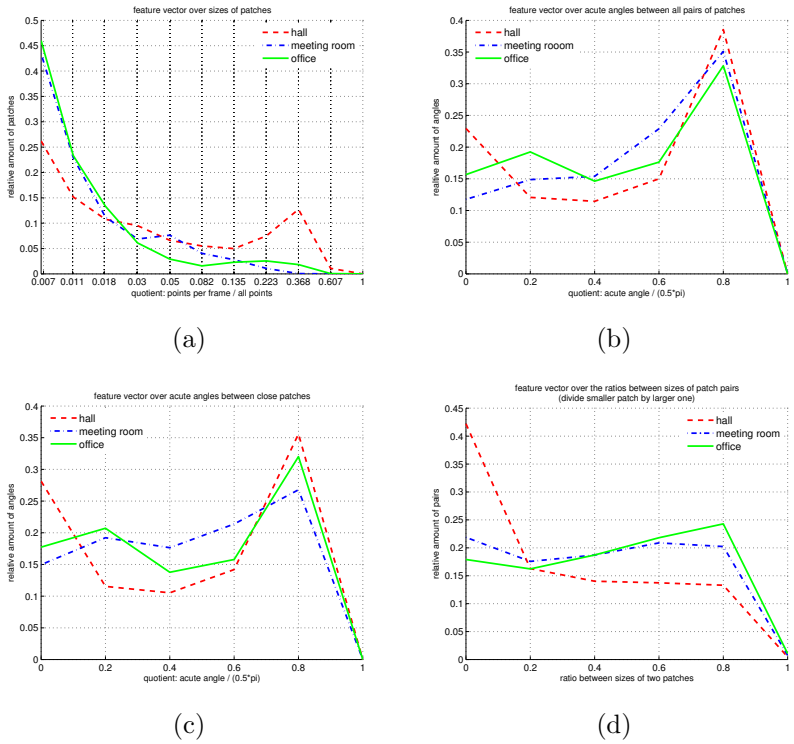


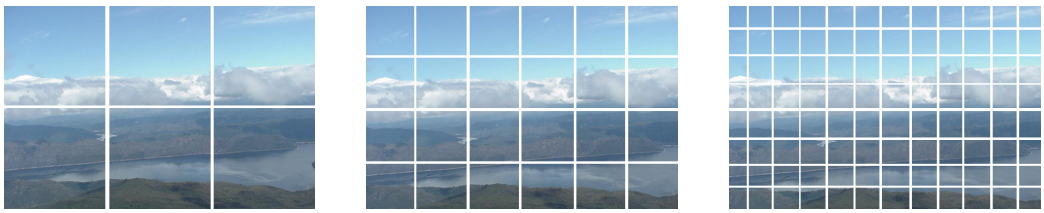
Figure 2.11: The histograms are computed over planes of 300 frames per room. Histogram over (a) the relative sizes of the planes, (b) all angles between pairs of patches divided by $\frac{\pi}{2}$, (c) angles between pairs of close patches divided by $\frac{\pi}{2}$ (d) the ratios between the size of two patches.

already seen rooms and around 80% for environments before unseen. The results are roughly stable for each of the different classifiers.

In the second experiment, the classification of individual rooms (6 different offices, 270 frames per office for training, 30 frames each for testing) is examined. Best results are reported for the NN-classifier (80% correct recognition rate). A room categorization that takes two offices with different layouts (1 person vs. 2 persons office) as training material achieves 88% correct categorization rate on unseen office environments.

2.2.2 Scenes as a configuration of parts

In the previous section, a scene was either characterized by ignoring local information or using absolute global positioning information that is determined



(a) (b) (c)

Figure 2.12: Coarse to fine partitioning of an image.

by a specific scene class. In the first case, features were integrated over large image portions or accumulated in feature histograms. In the second case, the image was either partitioned into a coarse regular grid or spatial templates were learnt from example images.

Relational representations

An alternative approach is to encode local information by relative spatial relationships between scene parts. Parts can but need not be defined on the level of objects. Frequently, objects cannot be reliably detected in complex scenes. In such cases, an image can be understood as a collection of *blobs* that are detected in a bottom-up segmentation process and may refer to an object part, complete object, collection of objects, or background.

Lipson (1996) stresses the importance of the spatial organization or scene structure for the classification of images. She argues for a scene representation as an organization of colored pixels rather than as an arrangement of objects. Her framework is based on relative spatial relationships between colored image regions. Both information is represented by qualitative terms. The spatial organization is based on a coarse to fine partitioning of the scene into equally sized square regions (Fig. 2.12). Each image region is described by the attributes: spatial position, color, luminance, and size. A scene is characterized by pairwise relationships between *salient* image regions regarding one of the given attributes. Salient regions are those which are discriminative for a specific scene class. The pairwise relationships are qualitatively described by '<', '>', '=' in the following attribute dimensions:

- x-, and y-coordinate (above, below, left, right). Overlapping x or y ranges are counted as equal (e.g. left or right).

- luminance is computed as the weighted sum of R,G,B with weights 0.3, 0.6, 0.1 (brighter, darker).
- color is treated along the R-,G-,B-dimensions as well as by computing cross channel relationships (e.g. blob B is perceptually blue: $B_b > B_g \wedge B_b > B_r$).
- size relationships are given by the coarse to fine partitioning of the image.

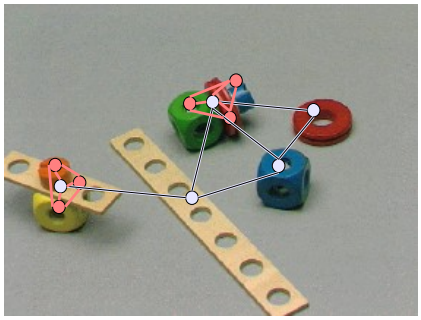
The qualitative model can be interpreted as a directed graph with blobs as nodes and relationships as edges. Thus, a match of the model and the image can be defined as a subgraph isomorphism. An alternative formulation would treat the scene model as a deformable template starting from a fixed spatial arrangement of averaged positions. Besides qualitative information the approach by Lipson also considers quantitative information, such as a minimum difference between related quantities or constant bounding conditions for selected attribute values.

The system was tested on a Corel dataset consisting of 7 themes ('Sunsets and Sunrises', 'Glaciers and Mountains', 'Coasts', 'California Coasts', 'Waterfalls', 'Lakes and Rivers') each containing 100 images. Using 4 handcrafted qualitative scene models with two to three nodes, Lipson achieves low false positive rates between 1-12%. False negative rates were significantly higher (20-67%).

Related approaches have been proposed by Smith & Chang (1995, 1996). In their system VisualSEEK, they present an image database indexing scheme where the image metric is based on image regions and spatial relationships. In a system query, the user can specify color, texture, size, and absolute position of several image regions on a grid. The system automatically extracts quantitative measurements from this description and uses a weighted combination of these cues for retrieval of similar images.

Other approaches concentrate on the evaluation of spatial relationships and neglect the object recognition problem (Abella & Kender, 1993).

Wachsmuth & Sagerer (2002) combine spatial relationships with an uncertainty model of the object recognition process as well as of the object and relation naming process. The nodes of the graph are defined by object detection results and are attributed with the object region, object type, and object color information from visual processing. Edges are defined by topological relations (e.g. connected *with*) or projective relations (e.g. *in front of*). Here,



Verbal scene specifications:

- "... bar with a bolt ..."
- "... motor in front of the bar ..."
- "... blue cube in front of the ring ..."
- "... long thing in the middle ..."
- etc.

Figure 2.13: Graph-based scene representations: a graph can be defined on different granularities and different relational semantics.

a partial scene model is specified by a verbal description. This needs to be matched to the visual representation. For this purpose a Bayesian network is dynamically constructed from the verbal and visual information given.

In Fig. 2.13 an example scene and possible verbal descriptions are given. The scenario is limited by a fixed number of elementary object types, a fixed number of color classes, and a uniform table background. However, more complex objects can be constructed by aggregation of elementary objects using bolts and nut-parts. These parts can dynamically be assigned names in the course of a dialog that refer to functional parts of the construction goal, e.g. "motor unit" of a toy-airplane. Thus, relations are defined on two different granularity levels: (i) relations between parts connected in an aggregated structure, (ii) relations between spatially separated scene entities. In a specific matching case, the level of granularity is selected by the wording used in the verbal description.

In Fig. 2.14 an example of the dynamically generated Bayesian network is shown. It is constructed from pre-modelled sub-networks that are combined with regard to the number of visual scene objects and the number of verbally described objects. The hidden variables IO and RO realize the mapping between verbal items and visual items. The mapping is constrained by the spatial relation that further depends on the reference frame selected by the speaker. The model parameters are partially learnt from labelled training

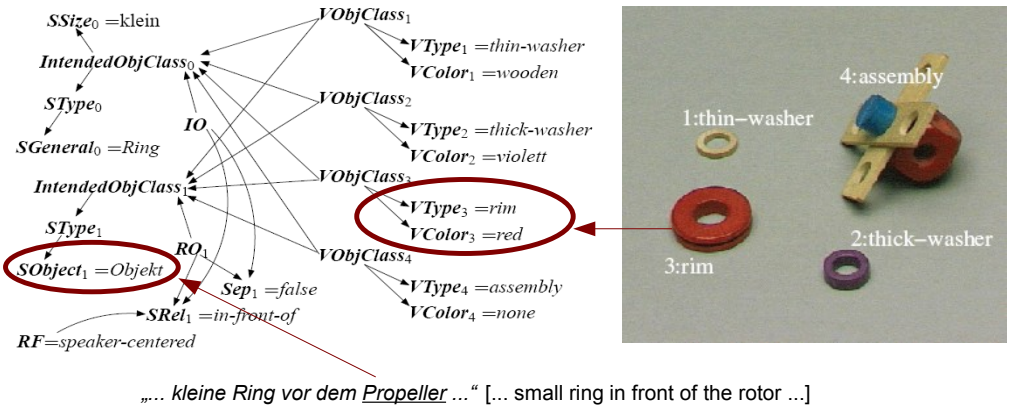


Figure 2.14: Example of a German dataset: The Bayesian network is dynamically generated from the visual and verbal information given. Note that the semantics of the word “Propeller” is previously not known. Therefore, a general term *object* is instantiated in the corresponding observable variable.

data[4] and are partially hand-crafted[5]. Further details of the Bayesian network is discussed in Sec. 2.3.4. The spatial model of the system is further described in Wachsmuth (2001).

The matching is computed by the maximum *a posteriori* hypotheses of the IO and RO variables:

$$(io^*, ro^*) = \underset{io,ro}{\operatorname{argmax}}\, Pr(IO = io, RO_1 = ro|\mathcal{E}) \tag{2.17}$$

where $\mathcal{E}$ is the set of observations given by the verbal description and the visual scene representation.

Evaluation experiments show that a correct scene identification can be inferred despite recognition errors on the visual as well as on the verbal side. The system achieves a correct object mapping (IO) (with two additionally selected objects allowed) of 76% for high input error rates of 21% lost or wrongly recognized verbal features as well as 15% false type classifications and 9% false color classifications on the visual side.

[4]On the vision side, the statistics of the object recognizer and pixel-based color classifier are measured by respective confusion matrices. The statistics on the class-specific wording is based on an online questionnaire.

[5]A hierarchy of nouns denoting object classes and super-classes has been modeled by hand.

Graph matching approaches

Relational representations encode the scene as well as the models as an attributed relational (or undirected) graph (ARG). The model graphs discussed so far have been rather small so that the graph matching problem does not cause complexity issues. In case of larger model and scene graphs, more sophisticated techniques for graph matching need to be applied. As an additional constraint, each matching approach also needs to deal with partial matches caused by possible occlusions. This raises the question of appropriate similarity measures.

Early work in has been conducted by Shapiro & Haralick (1985); Eshera & Fu (1986); Rosenfeld *et al.* (1976); Kittler & Hancock (1989). While Shapiro & Haralick and Eshera & Fu proposed similarity measures based on structure, Rosenfeld *et al.* and Kittler & Hancock introduced relaxation techniques that solve a maximum *a posteriori* (MAP) estimation problem by probabilistic evidential reasoning.

The former approach has been formulated in many variants based on discrete measurements, like edit distances (Eshera & Fu, 1986; Neuhaus & Bunke, 2006), as well as continuous measurements, like spectral graph theory Chung (1996) or graph embedding Demirci *et al.* (2006).

The latter approach has been further developed by Christmas *et al.* (1995) and Ahmadyfard & Kittler (2002) towards a systematic approach to combine observational evidence and *a priori* information about spatial context relations. The relaxation approach is tightly related to Markov Random Fields that have been applied to different computer vision problems that deal with spatial context (Geman & Geman, 1984; Li, 1995).

Set-of-features representations

A different approach is taken by the BlobWorld-system (Carson *et al.*, 1999) where each image is automatically segmented into regions (blobs). During this process (Fig. 2.15), pixels are assigned color, texture, and position features resulting in an 8D feature vector (x, y, L^*, a^*, b^*, contrast, anisotropy, polarity). Then, a Gaussian mixture model (with 2 to 5 mixture components) is fitted to the image data. Each Gaussian defines a cluster that is used for a connected component analysis which in turn generates a set of blobs. Blobs are described by a color histogram in the $L^*a^*b^*$ space with 218 valid bins. Additionally, the mean texture contrast and anisotropy is stored

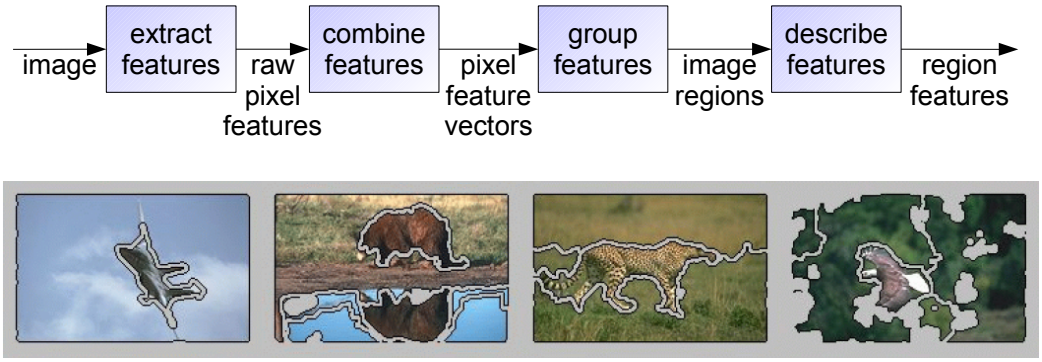


Figure 2.15: Blobworld processing stages and example image segmentations: from pixels to region descriptions (from Carson *et al.* (1999))

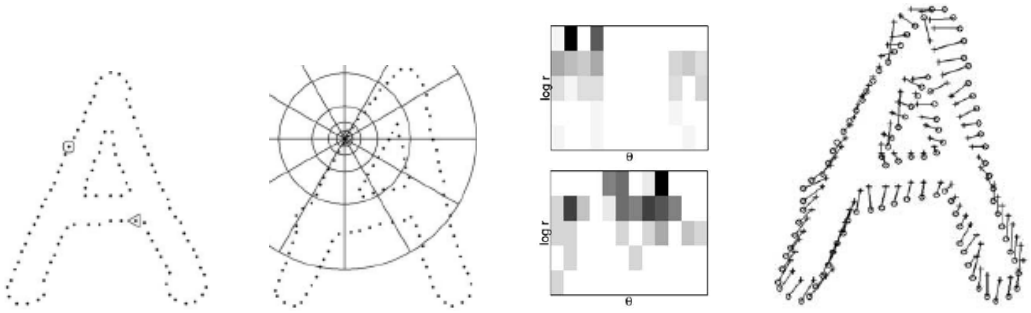
for each blob. Thus, an image is represented by a *set-of-features* approach that ignores local as well as relational information. However, a specific blob-match can still be localized in an image which provides a good transparency for possible user feedback. A drawback of this approach is the need for one-to-one correspondences. If a region is over- or under-segmented we loose possible matches and gain clutter.

An *atomic query* in Blobworld matches a single blob b_i with descriptor $\mathbf{v}_i$ of the query image to all blobs b_j with descriptor $\mathbf{v}_j$ of an image in the database. The similarity measure μ_{ij} is defined as follows:

$$\mu_{ij} = \exp\{-\frac{d_{ij}}{2}\} \text{ where } d_{ij} = (\mathbf{v}_i - \mathbf{v}_j)^T \Sigma (\mathbf{v}_i - \mathbf{v}_j). \tag{2.18}$$

Σ is a block-diagonal matrix. The block corresponding to the texture features is an identity matrix, weighted by a texture weight parameter. The block corresponding to the color features is defined by the quadratic distance of the cluster centers (mixture components), weighted by a color weight parameter.

A *compound query* score ("like-blob-1 and like-blob-2") is computed by applying fuzzy-logic operators. Images are ranked with regard to the overall score with the top-ranked presented to the user who is able to refine the query. The system was tested on 10 object categories (airplanes, black bears, brown bears, cheetahs, eagles, elephants, horses, polar bears, tigers, zebras) with 30 to 200 examples of each category among $10,000$ images in total. Carson et al. report for most of the categories that 2-blob-queries performed better than global color and texture histograms. However, the category 'planes'



(a) Model 'A' (b) Image 'A' (c) Shape descriptor (d) Bi-partite graph match

Figure 2.16: Shape context: The bi-partite graph is given by sets of nodes defined on the model 'A' and image 'A' (as depicted in (d)) that need to be matched. Each node is labeled by a shape descriptor that encodes a histogram over the distances and angles between the points on the shape boundary (shown in (b)). (c) depicts two exemplary shape descriptors from the boundary points marked in (a). In the work of Belongie et al. (2002) the matching costs of two nodes is defined by χ^2 distance between the corresponding histograms. The example has been taken from Belongie *et al.* (2002).

caused a performance drop because the small grey airplane region was not distinctive enough.

In blob-world, a model compound defined by a user query is composed out of a very few items or nodes. Each is described by a feature vector that is very discriminative with regard to the image class aimed for. Although the approach is built with one-to-one correspondences in mind, the fuzzy-logic operators do not enforce the constraint implied by it. It is possible that blob-1 and blob-2 are matched to the same image blob. Practically, this will be unproblematic as long as the user chooses distinctive query regions.

If features are weaker (less distinctive), matches need to be based on larger sets of feature nodes. In this case, it is more important to enforce the one-to-one correspondence constraint. The price to pay is an increased computational and space complexity. We need to solve a classical assignment problem in a bi-partite graph. The graph $G = (\mathcal{V}, \mathcal{E})$ is divided into two sets of nodes $\mathcal{P}, \mathcal{Q}$ with $\mathcal{P} \cup \mathcal{Q} = \mathcal{V}$. We define weighted edges $e = (p_i, q_i) \in \mathcal{E} = \mathcal{P} \times \mathcal{Q}$ between the two sets. Then, we search for the largest-size set of edges $\mathcal{S} \subseteq \mathcal{E}$ such that each node in the graph has a maximum width of 1. In case of two equally sized partitions, the node assignment given by $\mathcal{S}$ can be

described by a permutation $\pi = \{\pi(i)|i = 1 \dots |\mathcal{P}|\}$ of the indexed p-nodes $\mathcal{P} = \{p_1, \dots, p_i, \dots, p_{|\mathcal{P}|}\}$. Given the set of costs C_{ij} between all pairs (p_i, q_j), the algorithm has to minimize the total matching costs H,

$$H^* = \min_{\pi} H(\pi) = \min_{\pi} \sum_i C_{i,\pi(i)} \tag{2.19}$$

Fig. 2.16 shows an example for the *shape context* as defined by Belongie *et al.* (2002). Here, they circumvent a complex relational matching by summarizing relational information in a point descriptor. In this descriptor the spatial configuration of the set of localized feature points is coded by a histogram over pairwise distances and angles. The method has successfully been applied to the recognition of hand-written digits and 3D objects (using 100 points sampled from the Canny edges), shape retrieval from the MPEG-7 shape silhouette database (100 sample points, minimizing over normal, horizontally and vertically flipped models), and trademark retrieval (300 sample points, minimizing over an affine transformation model).

Another well established method for computing matching scores, that is based on an explicit assignment, is the Earth-Mover-Distance (EMD) as described by Rubner *et al.* (2000). The EMD defines a metric between two distributions. Instead of solving the matching problem in a bi-partite graph it solves the well-known *transportation problem* (Hitchcock, 1941). Here, we have a set of suppliers $\mathcal{P} = \{\mathbf{p}_i|i = 1 \dots |\mathcal{P}|\}$ and a set of consumers $\mathcal{Q} = \{\mathbf{q}_j|j = 1 \dots |\mathcal{Q}|\}$ that both have limited capacities $w_{\mathbf{p}_i}$ and $w_{\mathbf{q}_j}$, respectively. The transportation costs C_{ij} from each supplier i to each consumer j is given for a single unit of goods. The complete capacity of the suppliers and the consumers are normalized to 1. Then, we seek to find the least expensive flow of goods from suppliers to consumers. Intuitively, one can interpret the supplier distribution as a heap of earth and the consumer distribution as a hole. The least amount of work for moving the earth into the hole defines the score of the EMD, the corresponding flow of earth defines the solution. The EMD generalizes the bi-partite graph problem by the possibilities to assign larger chunks in one step and to assign fractions of units to target nodes. Thus, instead of localized single feature points p_i and q_j we deal with localized clusters $\mathbf{p}_i$ and $\mathbf{q}_j$ that have an associated capacity. Formally, we want to find a flow $\mathbf{F} = [f_{ij}]$, with f_{ij} defining the flow between the localized

clusters $\mathbf{p}_i$ and $\mathbf{q}_j$, that minimizes the overall cost H,

$$H^* = \min_{\mathbf{F}} H(\mathcal{P}, \mathcal{Q}, \mathbf{F}) = \min_{\mathbf{F}} \sum_{i=1}^{|\mathcal{P}|} \sum_{j=1}^{|\mathcal{Q}|} C_{ij} f_{ij} \tag{2.20}$$

A solution is subject to several constraints: (i) each single flow needs to be positive, $f_{ij} \geq 0$; (ii) the total flow from each supplier $\mathbf{p}_i$ needs to be covered by its capacity, $\sum_j f_{ij} \leq w_{\mathbf{p}_i}$; (iii) the total flow to each consumer needs to meet its capacity, $\sum_i f_{ij} \leq w_{\mathbf{q}_j}$; (iv) the total flow must be equal to the complete capacity of either supplier or consumer, $\sum_i \sum_j f_{ij} = \min\{\sum_i w_{\mathbf{p}_i}, \sum_j w_{\mathbf{q}_j}\}$. Given the optimal flow, the EMD is defined by the overall cost normalized by the total flow,

$$\mathrm{EMD}(\mathcal{P}, \mathcal{Q}) = \frac{\sum_i \sum_j f_{ij} d_{ij}}{\sum_i \sum_j f_{ij}} \tag{2.21}$$

The method has been applied to image retrieval using color distributions. Each image is transformed into the CIE-*Lab* color space and colors are clustered into fixed-sized bins (histogram) or individually defined bins (signature). In the second case, a cluster algorithm is run for each image separately. Independent of this choice, each cluster is represented by its average color $\mathbf{p}$ with weight or capacity $w_{\mathbf{p}}$. The image signature is defined by the set of pairs $(\mathbf{p}_i, w_{\mathbf{p}_i}), i = 1 \ldots N$ with N the average number of clusters between 8 and 40 depending on the application. For the pairwise comparison the L_2-Norm has been used. Belongie et al. report that other distance measures like χ^2 outperform EMD on coarse histograms but the EMD performs better on fine histograms or adaptive signatures. In coarse histograms the distance between clusters is quite large and becomes meaningless. They also did experiments including positional and texture information. In the case of positional features the pairwise distance is defined in a 5-dimensional space (adding the x,y-position of each pixel normalized in the range of 0 to 100).

Bag-of-features representations

Bag-of-features approaches have their roots in techniques for text retrieval. In the so called *bag-of-words* approach the text representation abstracts from the word ordering so that any permutation of the same text is treated identically. In probability theory, de Finetti (1990) has shown that this assumption corresponds to a mixture model for a collection of random variables

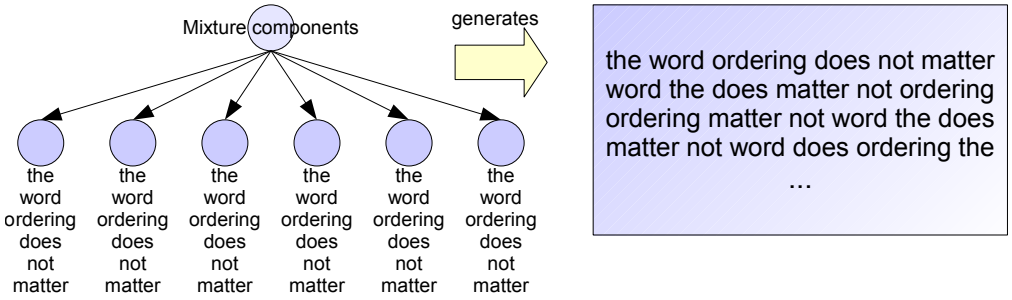


Figure 2.17: Bag-of-words assumption in text retrieval.

with a common cause that determines the mixture components of the words (Fig. 2.17). Note that in the probabilistic model there is no restriction that each word appears in the text with a fixed number. The generation process is stochastic in nature and will produce a text with a similar word frequency on average. For finding a similar document its vector of word frequencies is computed and the document with the closest vector is returned. Documents can be efficiently retrieved by using *inverted files*. In this indexing structure, each word of a corpus has an entry with a list of all documents (and positions in that document). the query document can be processed word by word keeping a voting statistics on the documents.

In Sivic & Zisserman (2003), this approach is transferred to the retrieval of objects and scenes in video. For scene matching, an entire frame is used as the query region. The task is to find the same location in other parts of the video. Each frame is represented as a bag-of-features. For this purpose, they compute two different types of interest point detectors:

- Shape Adapted (SA): The method computes interest points by an affine-adapted Harris detector (Mikolajczyk & Schmid, 2002). The scale is determined by the local extremum of a Laplacian operator across scale. The shape of an ellipse is optimized based on the maximization of the intensity gradient isotropy over the elliptical region.

- Maximally Stable (MS): The method is similar to computing maximally stable extremal regions (MSER). Therefore, a gray-level threshold is systematically increased from black to white leading to growing connected components similar to a watershed algorithm. Components that are maximally stable over a sufficient number of gray-levels are

approximated by fitting an elliptical shape (Matas *et al.*, 2002).

Each elliptical region is normalized by computing an affine transformation to a circle and is characterized by a 128-dimensional SIFT descriptor (Lowe, 1999). For each region the SIFT descriptors are averaged over 5 frames. Regions that cannot be tracked along a sequence of 5 frames are rejected in order to concentrate on re-detectable features of a scene, resulting in an average number of 1,000 regions per frame.

In order to describe a scene by a frequency vector of *visual words*, we need to define a *visual vocabulary*. Therefore, Sivic et al. compute a k-means clustering of SIFT descriptors using k values about 6,000 clusters for SA and 10,000 clusters for MS. A scene (document) d is represented by a k-vector $(t_1, \ldots, t_i, \ldots, t_k)^T$ of weighted word frequencies with

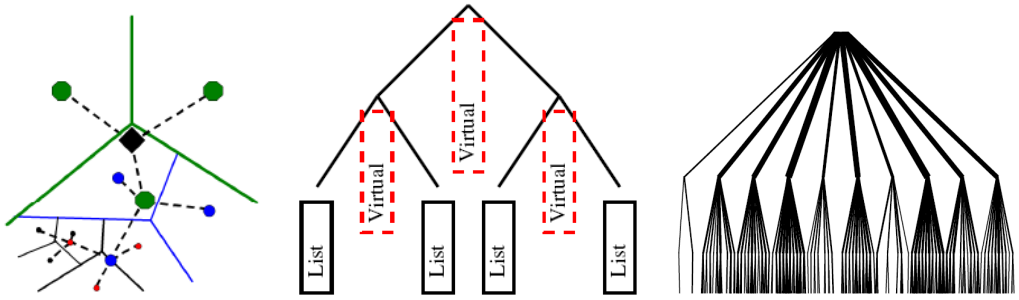
$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{n_i} \tag{2.22}$$

where n_{id} is the number of occurrences in document d, n_d is the total number of words in document d, n_i is the number of occurrences of term i in the whole database, and N is the number of documents.

The first factor (word frequency) weights words high that most accurately describe the document, while the second factor (inverse document frequency) down-weights words that frequently appear in the database.

The approach has been evaluated by Sivic et al. on the "Run Lola Run" movie achieving an average normalized rank (ANR) of relevant images around 0.013. The ANR is a number between 0 and 1.0 with 0.5 for random selections. 0.013 roughly means that if we search for a single relevant image in a set of 100 images, the relevant image is ranked between 2nd and 3rd place. Results can be improved using a *stop-list* which ignores the most frequent visual words (top 5%) and very rare visual words (bottom 10%). Another idea of improvement is to incorporate spatial consistency constraints, like restricting the local neighborhood for matches.

Nistér & Stewénius (2006) show that the bag-of-feature approach is scalable for image retrieval tasks in databases with up to 40,000 images. They propose a scoring scheme that can be efficiently evaluated in a vocabulary tree that is associated with an inverted file. Feature extraction is based on Lowe's SIFT descriptor computed on the set of Maximally Stable Extremal Regions (MSER) that are warped to circular patches. The vocabulary tree



(a) hierarchical k-means (b) indexing structure (c) image representation as a vocabulary tree

Figure 2.18: Vocabulary tree used for efficient indexing of bag-of-features representations. Images taken from Nistér & Stewénius (2006).

is constructed by a hierarchical k-means clustering of the SIFT descriptors with k defining the branching factor. Fig. 2.18(a) shows a visualization for $k = 3$ (typical values are around $k = 10$). The leaves of the resulting tree have attached inverted files for all features that are sorted into the branch while inner nodes only have virtual inverted files (Fig. 2.18(b)). An image is represented by the set of paths through the vocabulary tree. An example is shown in Fig. 2.18(c) with a branching factor of $k = 10$ and 400 features inserted from a single image. Formally, this is represented by a vector $\mathbf{d}$ for each database image,

$$\mathbf{d} = (d_1, \ldots, d_i, \ldots, d_L), \text{ with } d_i = m_i w_i \tag{2.23}$$

where m_i are the number of descriptors visiting node i, w_i is an associated weight, and L are the number of nodes in the vocabulary tree.

The weights model the relevance of a particular node for an image comparison and is typically set to an entropy weight $w_i = \ln(N/N_i)$ with N the number of images in the database and N_i the total number of images in the database that visited node i. The weights can be easily pre-computed for a given database. The term frequencies m_i are stored in the inverted files together with the image id-numbers. For an image comparison we need to compute the difference in L_p-norm between the normalized query image $\mathbf{q}$ and each database image $\mathbf{d}$,

$$||\mathbf{q} - \mathbf{d}||_p{}^p = \sum_i |q_i - d_i|^p \tag{2.24}$$

Splitting the sum into the zero and non-zero components of the image representation vector, the inverted file can be used in order to accumulate the score.

$$\begin{aligned}\sum_i |q_i - d_i|^p &= \sum_{i|d_i=0} |q_i|^p + \sum_{i|q_i=0} |-d_i|^p + \sum_{i|q_i\neq 0, d_i\neq 0} |q_i - d_i|^p \\ &= \sum_i |q_i|^p + \sum_i |d_i|^p + \sum_{i|q_i\neq 0, d_i\neq 0} (|q_i - d_i|^p - |q_i|^p - |d_i|^p) \\ &= ||q||_p^p + ||d||_p^p + \sum_{i|q_i\neq 0, d_i\neq 0} (|q_i - d_i|^p - |q_i|^p - |d_i|^p) \\ &= 2 + \sum_{i|q_i\neq 0, d_i\neq 0} (|q_i - d_i|^p - |q_i|^p - |d_i|^p)\end{aligned} \tag{2.25}$$

Given the single sum over the non-zero components, we can use the inverted files to add the contribution of a single feature to all relevant scores. Thus, by a traversing the vocabulary tree with the features of the query image we can compute all image comparisons in parallel. Query times on a database of 1 million images (frames from several movies) were reported about 1 second each.

Bag-of-feature approaches also have been extended with spatial information by image partitioning (Lazebnik *et al.*, 2006) and they are extensively used for object recognition tasks (Zhang *et al.*, 2005).

2.3 Using context in object recognition

The previous section concentrated on the categorization of scenes based on different representation schemes. Now, this information is exploited for further scene analysis.

An object $\mathbf{o}$ in an image can typically be described by a set of properties, like the label ω, the location in image coordinates x, the size or scale σ, etc. Given an image representation v, the object detection task can be probabilistically formulated by

$$Pr(O = \mathbf{o}|v) = \frac{Pr(v|\mathbf{o})}{Pr(v)} Pr(\mathbf{o}), \text{ with } \mathbf{o} = \{\omega, x, \sigma, \dots\} \tag{2.26}$$

The isolated object recognition approach assumes a conditional independence assumption that O is independent of the global image representation given a small image neighborhood B around x with size σ,

$$Pr(O = \mathbf{o}|v) \approx Pr(\mathbf{o}|v_L) = \frac{Pr(v_L|\mathbf{o})}{Pr(v_L)} Pr(\mathbf{o}); \quad v_L = v_{B(x,\sigma)} \tag{2.27}$$

As also suggested by Torralba (2003) we can extend this approach by dividing the image representation v into two feature sets:

$$v = \{v_{B(x,\sigma)}, v_{\overline{B}(x,\sigma)}\} = \{v_L, v_C\} \tag{2.28}$$

With Bayes' rule we can rewrite Eq. 2.26 as

$$\begin{aligned} Pr(O = \mathbf{o}|v) &= \frac{Pr(\mathbf{o}, v)}{Pr(v)} = \frac{Pr(v_L|\mathbf{o}, v_C) Pr(\mathbf{o}|v_C) Pr(v_C)}{Pr(v_L|v_C) Pr(v_C)} \\ &= \frac{Pr(v_L|\mathbf{o}, v_C)}{Pr(v_L|v_C)} Pr(\mathbf{o}|v_C) \end{aligned} \tag{2.29}$$

The first factor describes the contextual influence on the appearance of the object (global illumination, pose of the object), the second factor describes a priming of certain object classes by context. Most work concentrates on the second factor (contextual prior). The challenge is to deal with the high dimensionality of v_C. In terms of O, most approaches concentrate on object category, image location and scale.

$$Pr(O = \mathbf{o}|v_C) = Pr(\sigma|x, \omega, v_C) Pr(x|\omega, v_C) Pr(\omega|v_C) \tag{2.30}$$

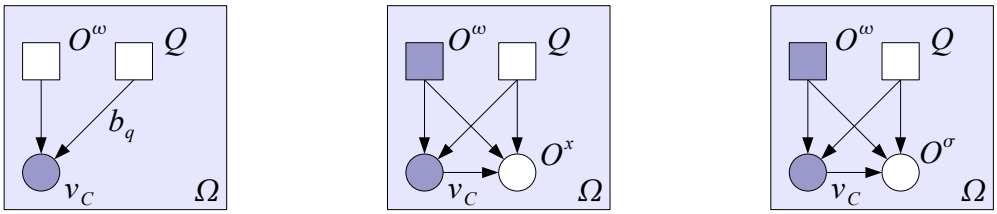
The three factors are *object priming, focus of attention, scale selection.*

Alternatively, we can start from a collection of already detected object hypotheses that are denoted by the random variables $O_1, \ldots O_K$ and look at a globally optimized feature and class assignment,

$$\begin{aligned} (\hat{\mathbf{o}}_1, \ldots, \hat{\mathbf{o}}_K) &= \underset{\mathbf{o}_1,\ldots,\mathbf{o}_K}{\operatorname{argmax}} Pr(O_1 = \mathbf{o}_1, \ldots, O_K = \mathbf{o}_K|v) \\ &\approx \underset{\mathbf{o}_1,\ldots,\mathbf{o}_K}{\operatorname{argmax}} Pr(O_1 = \mathbf{o}_1, \ldots, O_K = \mathbf{o}_K|v_L^{(1)}, \ldots v_L^{(K)}) \end{aligned} \tag{2.31}$$

where $v_L^{(k)}$ are the potential local features of the object hypothesis O_k.

In this case, the structure of the probabilistic model is determined by the local interactions between the object hypotheses (e.g. occlusions) and the corresponding object models (e.g. feature dependencies).



(a) object priming (b) focus of attention (c) selection of scale

Figure 2.19: Graphical models for estimating object detection parameters from holistic scene representations. Ω is the number of different object classes. Rectangles are discrete variables, circles are continuous variables, O^ω is a binary variable being true for a positive object class detection, Q is a mixture variable for Gaussian distributions, O^x denotes the location, and O^σ the scale of the object.

2.3.1 Combining holistic context and object detection

Torralba (2003) presents an approach based on the first scheme. He estimates an object priming, a focus of attention, and a scale selection from a holistic scene characterization v_C (gist) as described in Sec. 2.2.1. Fig. 2.19 shows the corresponding graphical models of these three estimation tasks.

Object priming factor

Using Bayes' theorem the priming factor $Pr(O^\omega = \omega|v_C)$ is transformed into

$$Pr(\omega|v_C) = \frac{Pr(v_C|\omega)Pr(\omega)}{Pr(v_C|\omega)Pr(\omega) + Pr(v_C|\overline{\omega})P(\overline{\omega})} \tag{2.32}$$

where $\overline{\omega}$ refers to images that do not include an object class ω. The probability $Pr(v_C|\omega)$ is approximated by a Gaussian mixture distribution

$$Pr(v_C|\omega) = \sum_q b_q \mathcal{N}(v_C; \mu_q, \Sigma_q) \tag{2.33}$$

The parameters can be learnt from a labeled training set using an EM-style algorithm. The *a priori* probability of finding an object o in an image is set to $Pr(\omega) = 0.5$.

Focus of attention

The focus of attention $Pr(x|\omega, v_C)$ is modeled assuming a linear dependency between the variables x and v_C,

$$Pr(x|v_C, \omega) = \frac{\sum_q b_q \mathcal{N}(x; x_q, X_q)\mathcal{N}(v_C; v_q, V_q)}{\sum_p b_p \mathcal{N}(v_C; v_p, V_p)}, \text{ with } x_q = a_q + A_q(v_C - v_q). \tag{2.34}$$

The model parameter are $(b_q, a_q, A_q, X_q, v_q, V_q)_{q=1\ldots M}$ with a typical choice of $M = 4$. These can be learnt given a random set of images with occurrence of ω and labeled bounding boxes corresponding to the object areas.

Scale selection

The selection of scale $Pr(O^\sigma = \sigma|x, \omega, v_C) \approx Pr(\sigma|\omega, v_C)$ is modeled similar to the focus of attention, but ignores the dependency on the locality x,

$$Pr(\sigma|\omega, v_C) = \frac{\sum_q b_q \mathcal{N}(\sigma; \sigma_q, S_q)\mathcal{N}(v_C; v_q, V_q)}{\sum_p b_p \mathcal{N}(v_C; v_p, V_p)}, \text{ with } \sigma_q = a_q + A_q(v_C - v_q). \tag{2.35}$$

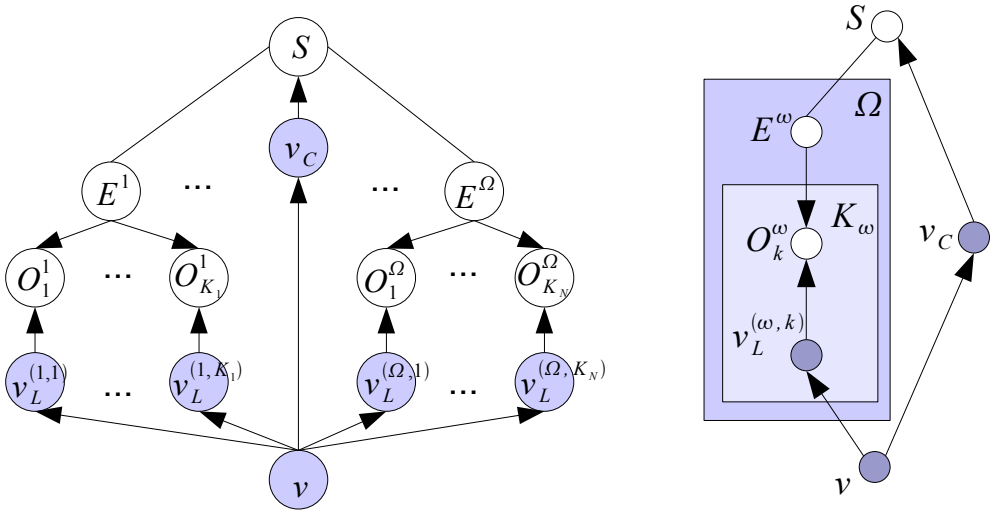
Here the model parameters are $(b_q, a_q, A_q, S_q, v_q, V_q)_{q=1\ldots M}$.

A joint probability model

Murphy *et al.* (2003) propose a joint probability model connecting the classification of scenes with object detection results. The graphical model is depicted in Fig. 2.20. It encodes the following joint conditional density,

$$Pr(S, E^{1:\Omega}, O^1_{1:K_1} \ldots O^\Omega_{1:K_\Omega}|v) = \frac{1}{Z} Pr(S|v_C) \prod_{\omega=1}^{\Omega} \phi(E^\omega, S) \prod_{k=1}^{K_\omega} Pr(O^\omega_k|E^\omega, v_L^{(\omega,k)}) \tag{2.36}$$

where E^ω is a binary variable modeling the existence of an object class ω in the image. $Pr(S = s|v_C)$ is the output of the s-vs-other scene classification, $\phi(E^\omega, S = s)$ is a table which counts the number of times object type ω occurs in scene type s. O^ω_k is a binary variable that validates the k-th detection hypothesis for object class ω on the image patch $v_L^{(\omega,k)}$. $Pr(O^\omega_k|E^\omega, v_L^{(\omega,k)})$ models the joint influence of the object detector and scene classification on object recognition results.



(a) graphical model (b) plate notation

Figure 2.20: Graphical model for scene and object recognition. $\Omega = 6$ is the number of object classes, $K_\omega \approx 5000$ is the number of patches for class ω.

$Pr(O_k^\omega | E^\omega, v_L^{(\omega,k)})$ is defined as follows

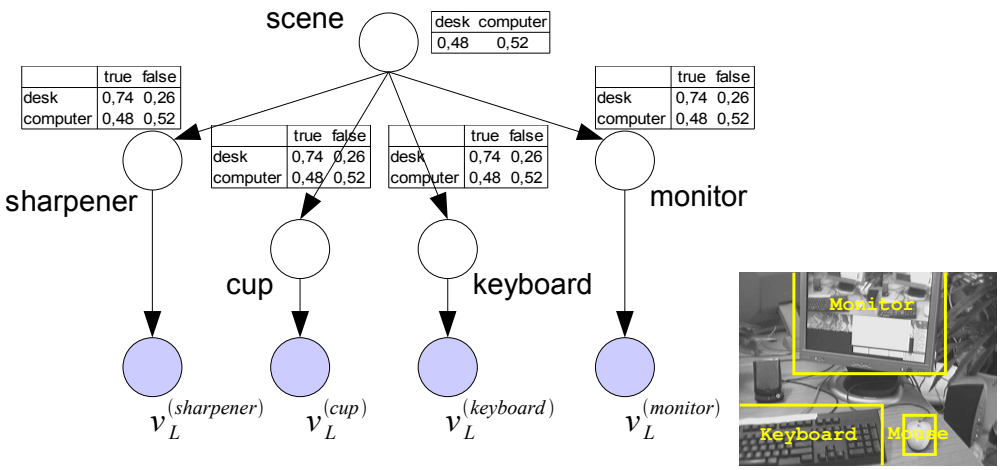
$$P(O_k^\omega = 1 | E^\omega = e, v_L^{(\omega,k)}) = \begin{cases} \sigma(w^T[1 \alpha^\omega(v_L^{(\omega,k)})]) & \text{if } e = 1 \\ 0 & \text{if } e = 0 \end{cases} \quad (2.37)$$

where $\alpha^\omega(v_L^{(\omega,k)})$ is the confidence of the boosted object classifier for class ω on the image patch $v_L^{(\omega,k)}$ and $\sigma(w^T[1\alpha])$ is the conversion into a probability using logistic regression ($\sigma(x)$ is the sigmoid function).

Intuitively, the formula suppresses object detections of class ω, if ω is incompatible with the scene and otherwisely uses the bottom-up object detection results.

2.3.2 Detecting semantic object-scene inconsistencies

Frequently, scene categories are defined by functional rather than appearance-based aspects. A simple way of defining the functionality of a scene setup is based on the object types typically found in a scene. An



(a) Bayesian network

(b) Example computer scene

Figure 2.21: Semantic model for two different scene contexts ("desk", "computer"). Both are defined by the presence of relevant key objects.

example is given in Fig. 2.21. Here, a Bayesian network was defined for modeling different functional scene contexts (Bauckhage *et al.*, 2008a; Hanheide, 2006). A computer working place consists of a monitor and keyboard, a writing desk may have a sharpener and cup placed on. The conditional probability tables have been estimated on short video sequences of each scene type. The conditional probabilities for the existence of an object type, e.g. $Pr(v_L^{(cup)}|cup)$, are given by a confidence value of the corresponding object detector transformed into a probability. The Bayesian model can be read in two directions. It can be used in a diagnostic way by computing the belief of a scene type, or as a consistency model for object recognition results. In the second case, a conflict measure can be defined similar to an *emergence measure* defined in Jensen & Nielsen (2007).

$$conf(\mathbf{e}) = \log_2\left(\frac{1}{Pr(\mathbf{e}|\mathcal{M})}\prod_i Pr(e_i)\right) \tag{2.38}$$

where $\mathbf{e} = \{e_1, \ldots, e_K\}$ is the set of observations or evidences, $\mathcal{M}$ is the probabilistic scene model, $Pr(\mathbf{e}|\mathcal{M})$ can be computed by propagating the evidences in the network.

This provides a summarized measurement if the observations are consistent with the assumptions made by the model. Negative values of *conf*(.) indicate

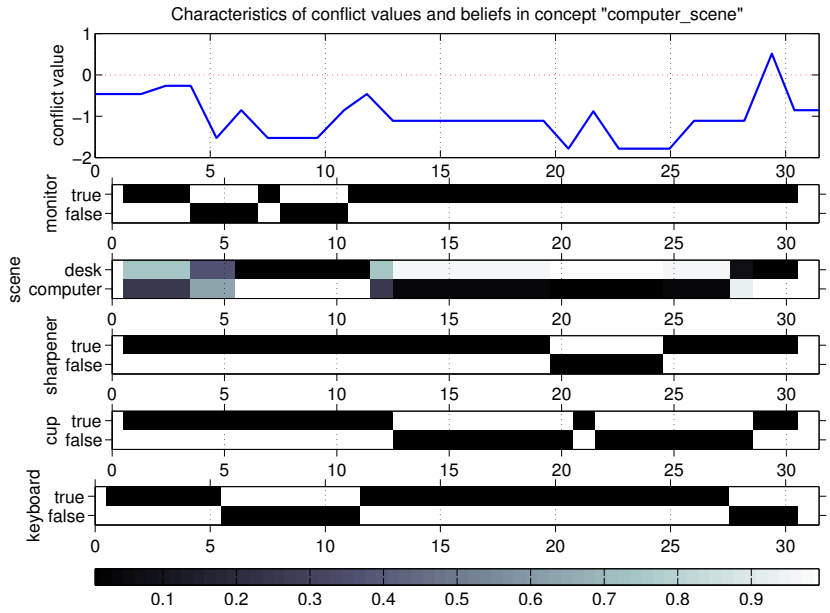


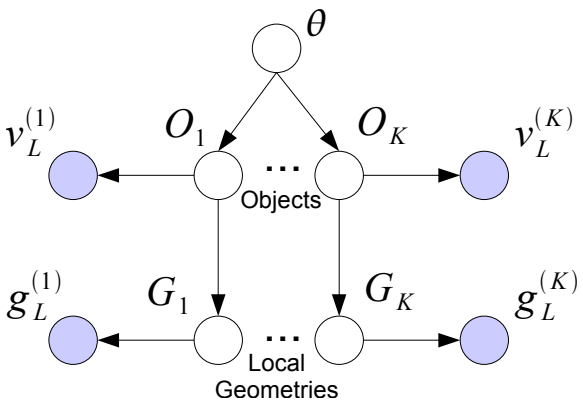
Figure 2.22: Results for field of view context classification. The graph depicts the states of the variables of the Bayesian network in Fig. 2.21 over a video sequence. White denotes a belief of 1.0, black a belief of 0.0. The conflict value is computed based on equation 2.38.

consistent evidence, positive values indicates either invalid recognition results or an invalid model. Hints on invalid hypotheses can further be gained by comparing the causal and diagnostic support of the different object hypotheses nodes (“sharpener”, “cup”, etc.)

In Fig. 2.22 the scene belief and conflict values for a labeled video sequence is shown. One can observe that the conflict value raises when the observed scene changes from desk to computer and vice versa.

2.3.3 Understanding objects in 3D scenes

Objects are entities placed in the 3D space rather than 2D image plane. As a consequence, there are several contextual relationships related to the 3D geometry of scenes that can be exploited for object recognition purposes. In the approaches discussed in the following, scenes are mostly understood as a collection of objects. Thus, there is an interplay between local scene properties, like occluded object parts, surface orientations, or depth.



(a) Graphical model

(b) Example scene

Figure 2.23: Graphical model for considering local scene geometries. (b) shows an example for a confidence map of the surface class "ground". The blue line is the estimated horizon line. Finally detected cars are marked by green boxes. The images have been taken from Hoiem *et al.* (2006).

Utilizing local geometries

Local geometries imply many constraints on consistent scene objects. Objects placed in a similar depth appear on a similar scale, objects are not hovering in the air but are attached to common ground surfaces. Hoiem *et al.* (2006) present a probabilistic model for the interplay between object detectors, rough scene geometry, and the approximate camera viewpoint (determined by camera height and tilt angle) that is applied to a single 2D image.

The corresponding graphical model is presented in Fig. 2.23. It encodes the following decomposition of the conditional joint probability

$$\begin{aligned} &Pr(\theta, O_{1:K}, G_{1:K} | v_L^{(1:K)}, g_L^{(1:K)}) \\ &\quad \propto Pr(\theta) \prod_k Pr(O_k|\theta) \frac{Pr(O_k|v_L^{(k)})}{Pr(O_k)} P(g_L^{(k)}|O_k) \frac{Pr(G_k|g_L^{(k)})}{Pr(G_k)} \end{aligned} \tag{2.39}$$

The viewpoint θ is modeled by two variables assuming a single ground plane where the relevant objects are placed on,

$$Pr(\theta) = Pr(v_0) Pr(y_c) \tag{2.40}$$

where v_0 is the vertical position of the horizon, y_c is the camera height related to the ground plane. Both independent probability distributions are modeled by a Gaussian estimated from a set of training images. y_c is not directly observable, but can be computed from the scene objects with known height by the relation $y_k/y_c = h_k/(v_k - v_0)$ where y_k is the 3D object height, v_k is the bottom position, and h_k is the height of the image object.

An object detector provides a class-conditional log-likelihood c_k at each position and scale (with discrete steps). Thus, the probability of finding an object O_k at a particular image patch $v_L^{(k)}$ is given by

$$P(O_k|v_L^{(k)}) = 1/(1 + \exp[-c_k - \log \frac{Pr(O_k)}{1 - Pr(O_k)}]) \tag{2.41}$$

The object height h_k depends on the viewpoint θ giving

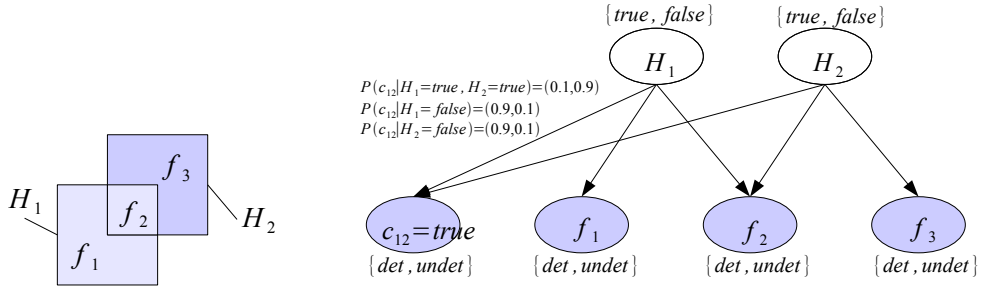
$$\begin{aligned} Pr(O_k|\theta) &= Pr(\omega_k, u_k, v_k, h_k, w_k|\theta) = Pr(h_k|\omega_k, u_k, v_k, w_k, \theta) Pr(\omega_k, u_k, v_k, w_k|\theta) \\ &\propto Pr(h_k|\omega_k, v_k, \theta) \end{aligned} \tag{2.42}$$

where ω_k is the object class and (u_k, v_k, h_k, w_k) defines the bounding box of an object hypothesis O_k.

The variables besides h_k are assumed to be equally distributed with regard to θ. $P(h_k|\omega_k, v_k, \theta)$ is modeled as a Gaussian with parameters transformed from those of the class specific 3D object height y_k. If the distribution of the 3D object height is described by a Gaussian $\mathcal{N}(\mu, \sigma)$ the parameters for h_k are given by $\frac{\mu y_k(v_0 - v_k)}{y_c}$ and $\frac{\sigma y_k(v_0 - v_k)}{y_c}$.

Hoiem et al. use the method of Hoiem *et al.* (2005) in order to compute confidence maps for three main surface classes ("ground","vertical","sky") and five surface subclasses of vertical (planar facing "left","right", and "center", and non-planar "solid" and "porous"). The variable G_k has three possible values "no object surface", "object surface", and "object surface with ground below" that refer to the object detection window. $Pr(G_k|O_k)$ is computed by counting occurrences on a training set and $Pr(g_L^{(k)}|G_k)$ is estimated based on the confidence maps.

The approach has been successfully tested on images from busy city streets, sidewalks, parking lots, and roads which provide enough depth for horizon detection and have a single ground plane.



(a) Example of occlusion (b) Bayesian network

Figure 2.24: Representing mutually exclusive hypotheses using a dummy constraint node.

Modeling occlusions

Westling & Davis (1998) define an interpretation as a collection of hypothesized objects together with their 3D pose. They use a separate method (Westling & Davis, 1996) in order to generate hypotheses of objects from images and a set of models. A feature vector is associated with an image region and encodes both colors and edges. Each hypothesis is associated with one list of matched features (together with a degree of match) and one list of predicted but unmatched features. Hypotheses are rated by internal factors (degree of match between image and entire interpretation) and external factors (background knowledge about the compatibility or incompatibility of object classes).

The Bayesian network encodes causal relationships between hypotheses and image features as well as visual and physical relationships in between hypotheses. Hypotheses are represented as unobserved binary nodes H_k with *true / false* values. The prior probability is uniformly set to 0.5. Image features are represented as observed nodes f_i with two values, one encoding that the feature predicted by a hypothesis was detected and one encoding its absence (or presence of a different feature at the same location). Object hypothesis nodes are connected to those feature nodes that they predict (causal relationship). In case a feature location is shared by two object hypotheses (H_i, H_j), either the hypotheses are competing because they are not allowed to occupy the same space or one object is occluding the other. The first case is modeled by a dummy constraint node c_{ij} as depicted in

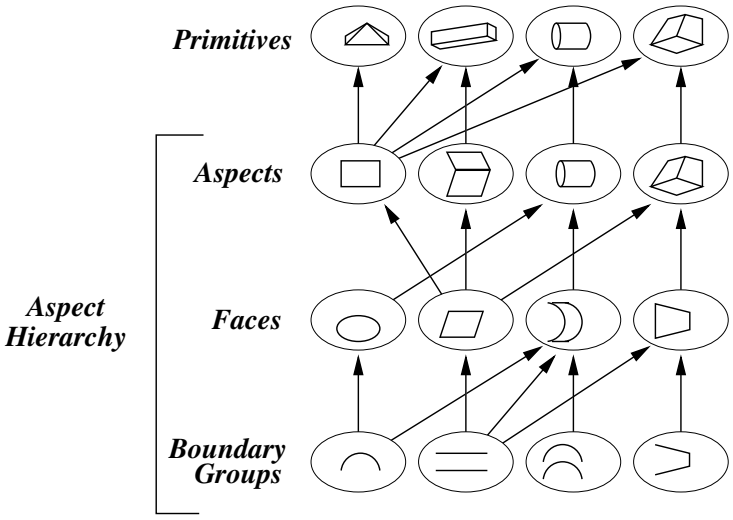


Figure 2.25: The aspect hierarchy for recognizing primitives (Dickinson *et al.*, 1992).

Fig. 2.24(b). The second case is shown in Fig. 2.24(a). The structure of the Bayesian model would still be the same except the dummy node c_{12} would be missing. The features f_2 are assigned to the occluding hypothesis H_1, but the relation to feature f_2 still depends on the occluded hypothesis H_2. If H_1 is true and f_2 does not match the prediction of the occluded hypothesis, it would still support H_2 because the mismatch is explained by H_1.

Modeling aspect hierarchies

Westling and Davis realize a top-down verification procedure for 3D object hypotheses. If we want to use 3D spatial constraints earlier in the grouping process we need to estimate 3D shapes from low level features in order to infer potential consequences in 3D space. Dickinson *et al.* (1992) present the OPTICA system that employs Bayesian networks in a 3D shape recovery approach. An object recognition task is divided into the finding of 3D volumetric primitives that could be used for object indexing and the recognition of complex objects that can be constructed by combining these primitives. The idea of this approach is to extract a small finite set of powerful indexing primitives in order to access a large, possibly infinite, object database. The computational burden is, therefore, partially shifted from top-down verification of simple 2D features towards the bottom-up extraction and grouping

of 2D features into volumetric primitives.

The Bayesian network approach is used to constrain the search of the grouping process. Given an arbitrary set of 3D primitives, an aspect hierarchy is constructed that realizes a probabilistic mapping from 2D features to these volumetric primitives. In order to account for occlusions in the scene, the aspect hierarchy is organized in levels of different complexity (Fig. 2.25):

- *Aspects* constitute the top level of the hierarchy. They represent the set of topologically distinct views of the primitives.

- *Faces* correspond to the primitive surfaces. Combinations of them define the aspects.

- *Boundary groups* represent all subsets of lines and curves comprising the faces. They define the lowest level of the aspect hierarchy.

All elements of the aspect hierarchy *qualitatively* represent geometric elements of the image. Boundary groups and faces are defined by a graph of qualitative relationships (*intersection, parallelism, symmetry*) among qualitatively described contours, e.g. two parallel lines of equal length is a boundary group that is a subgraph of several face graphs. Aspects denote graphs in which nodes represent faces and edges represent face adjacencies.

The hierarchy consists of 37 different aspects, 16 faces, and 31 boundary groups. The ambiguities in the mapping from a lower level to the next higher level are captured by conditional probability tables:

$$Pr(primitive|aspect), Pr(aspect|face), Pr(face|boundary\text{-}group). \qquad (2.43)$$

The CPTs are estimated from simulated data. The primitive volumes are rotated around their internal axes and projected onto the image plane. The appearance of each feature and its parent is noted and counted. A priori probabilities of occurrence or orientation of primitives can be considered during the simulation process.

The Bayesian network that is defined by the three conditional probability tables is exploited during shape recovery in the following way: First, a contour graph is calculated in which nodes denote curvature discontinuities or junctions of contours and in which edges are the actual bounding face contours. From this graph, closed image faces are extracted. If the image

face exactly matches a face of the aspect hierarchy this is directly used as an evidence:

$$\underline{e}_{face}(i) = \begin{cases} 1.0, & \text{if } face\ i \text{ has exactly been matched} \\ 0.0, & \text{otherwise.} \end{cases} \tag{2.44}$$

If there is no exact match the Bayesian network is instantiated on a lower level using the boundary groups found for the image face:

$$\underline{e}_{boundary\text{-}group}(i) = \begin{cases} 1.0, & \text{if } boundary\ group\ i \text{ has been matched} \\ 0.0, & \text{otherwise} \end{cases} \tag{2.45}$$

resulting in a probability distribution of face labels. Then the probability of the most probable explanation of an *aspect* hypotheses a is calculated that might encompass that face. Either the evidence $\mathbf{e} = \underline{e}_{face}$ or $\mathbf{e} = \underline{e}_{boundary\text{-}group}$ is given:

$$Pr(aspect = a|\mathbf{e}) = \max_{b,f} Pr(aspect = a, face = f, boundary\text{-}group = b|\mathbf{e}) \tag{2.46}$$

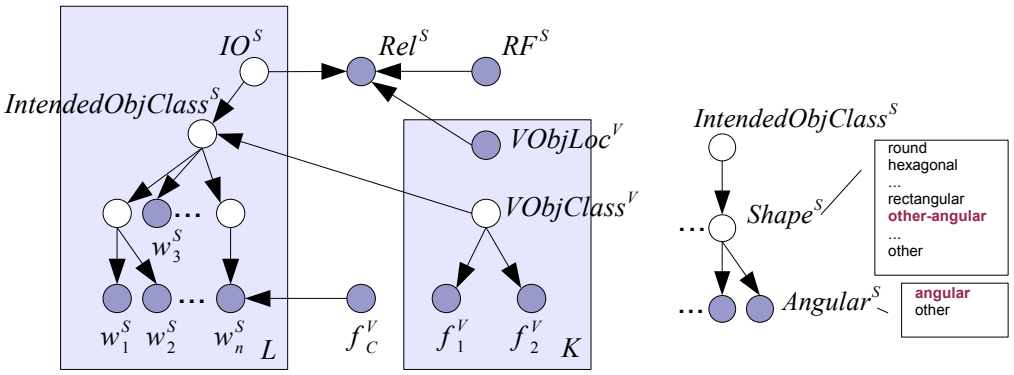
This probability is used in order to constrain the search for an aspect instantiation, i.e. the finding of a graph match of an aspect from the aspect hierarchy with a subset of image faces. In the same way, the search process for the primitives can be constrained by the aspects found:

$$Pr(primitive = p|\underline{e}_{aspect}) \tag{2.47}$$

During the face labeling process, the aspect hierarchy and the conditional probabilities defined in it can be additionally exploited in order to get rid of segmentation errors. For this purpose, the authors present a model-based region merging algorithm. Starting with an over-segmentation, the algorithm merges two faces if the probability of a face label can be increased.

2.3.4 Integrating visual and verbal object descriptions

In Sec. 2.2.2 we already discussed how verbally specified scene descriptions can be matched to visual representations. The same probabilistic framework can be used in order to improve object recognition results. For this purpose we take a closer look at the model. The generic graphical model is shown



(a) Generic graphical model (b) Shape abstractions

Figure 2.26: Graphical model for the integration of verbal and visual information. Verbal information variables are marked by a superscript ^{S}, visual information by a superscript ^{V}. The verbal description is given by L noun phrases related by a named relation Rel^S. In case of spatial relations, words like "left-of" are chosen with regard to a reference frame RF^S. A noun phrase includes verbal features $w_{1:n}^S$ like object nouns or *color* and *shape* attributes. K denotes the number of objects in a scene with extracted features $f_{1:m}^V$. The variable $IO \in \{1, \ldots, K\}$ models the mapping between noun phrases and scene objects. In (b) the value "angular" is modeled as an abstraction of the "hexagonal", "rectangular", and "other-angular" states.

in Fig. 2.26. Four different problems have to be solved by it: (i) Words like object nouns or adjectives have to be translated into visual object classes, (ii) verbal noun phrases have to be mapped to objects detected in an image, (iii) the mapping of noun phrases and scene objects needs to be consistent with relational information given by a speaker, (iv) the global scene context partially influences the semantics of words, e.g. "long" would be used for different object types dependent on the length of other objects in the scene.

Translating words into object classes: In Wachsmuth & Sagerer (2002) and Socher *et al.* (2000) conditional probability tables for $P(w^S|IntendedObjClass^S)$ are estimated from a WWW-questionnaire. However, given an attribute dimension like *color* or *shape* words define attribute values on different abstraction levels. For example, "colored" may refer to different saturated colors like "red", "green", "blue", "yel-

low". Vorwerg (2001a,b) even found ambiguous abstraction levels for words like "angular" that either contrast with other angular shapes like rectangular or abstracts from specific angular variants.

Mapping noun phrases to scene objects: The variable IO^S selects between different scene objects. The corresponding conditional probability table is modeled as follows:

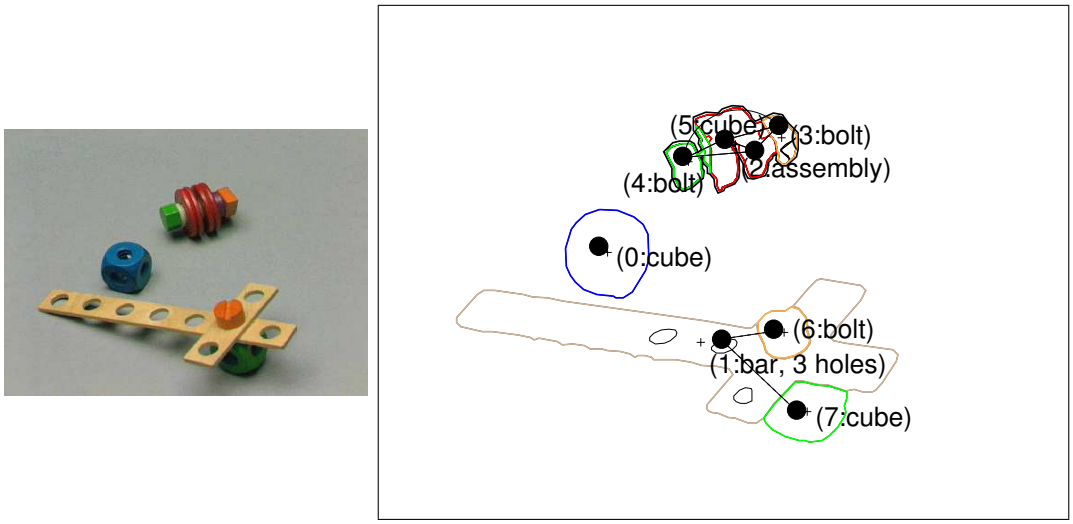
$$\begin{aligned} &Pr(IntendedObjClass^S | IO^S, ObjClass^V_{1:K}) \\ &= Pr(IntendedObjClass^S | ObjClass^V_k) = Id, \text{ if } IO^S = k. \end{aligned} \tag{2.48}$$

where Id is the identity matrix.

Exploiting relational information: The probable selections of the variables $IO^S_{1:L}$ are restricted by a relation Rel^S. If only one noun phrase is specified ($L = 1$) this could be a local description like "on the left". Typical cases include two noun phrases ($L = 2$) that are related by a projective spatial relation like *object-A* "left of" *object-B*. In some cases a relation can also be defined between three noun phrases like "between". The probability $Pr(Rel^S | IO^S_{1:L}, RF^S)$ is generated from a computational spatial model. The reference frame RF^S can either be set by default, be specified verbally ("from my perspective"), or be inferred from the other evidences.

Context-dependent word semantics: The variable f^V_C encodes a global feature of the scene that fixes the frame of reference for attributes describing the object type. For example, the longest object in the scene influences the semantic scale of the size dimension, so that the words "long", "middle-long", and "short" may refer to different object types.

An example for exploiting verbal information for object recognition is presented in Fig. 2.27. Here the two *red rims* of the top aggregated structure as well as the *orange rhomb-nut* tight to it are erroneously classified as *red cube* and *orange bolt*. Nevertheless, the relational information given by "with" is strong enough to bind the noun phrases to the right scene objects ($IO = 5$ for the first noun phrase). Using the verbal features the probabilistic model correctly infers the object class *red rim* for scene object 5.



speaker: *"... den roten Ring mit der Raute." [... the red ring with the rhomb.]*

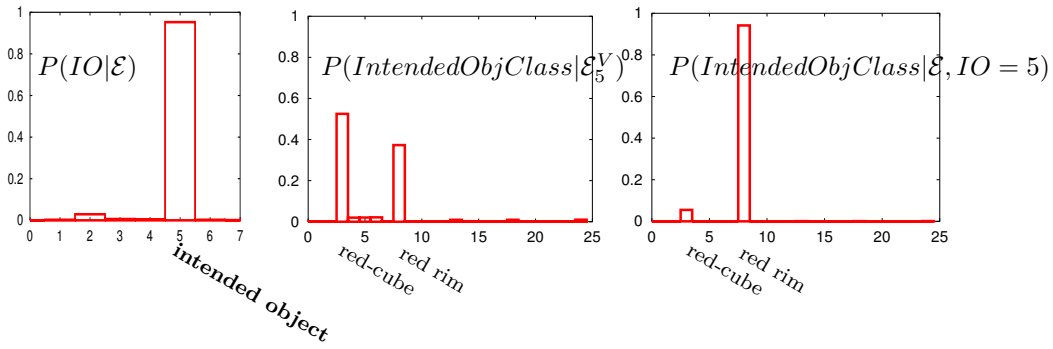


Figure 2.27: Example: The speaker refers to object 5 and reference object 3. The rim (obj. 5) and the rhomb-nut (obj. 3) are mis-recognized: $\mathcal{E}_5^V = \{red,\ cube\}, \mathcal{E}_3^V = \{orange,\ bolt\}$. The mounting relation between them is correctly detected (solid lines). The dotted lines in the lower part represent relations between touching object regions. The evidences extracted from speech are $\mathcal{E}_0^S = \{\text{rot, Ring}\}, \mathcal{E}_1^S = \{\text{mit, Raute}\}$. $\mathcal{E} = \bigcup_{i=0...7} \mathcal{E}_i^V \cup \mathcal{E}^S$. The system correctly selects object 5 with reference object 3. The class of object 5 *rim* is correctly inferred.

2.4 Summary and conclusion

The consideration of context in computer vision is a frequently re-occurring topic especially for the task of object recognition. In this chapter, I have motivated what might be gained from it and I have explored various techniques to do it. However, it remains a patch-work if one tries to look at the overall picture. Parts of it are nicely covered by homogeneous frameworks – mostly utilizing probabilistic graphical models – that fuse object recognition results with contextual cues. These can be roughly grouped into global and local cues. Global cues ignore relational structure and collect summarizing scene statistics. These gain their performance from the sheer mass of support. Local cues need to solve the correspondence problem when matching models and scenes. As a consequence, the model size is typically very small because of complexity issues. The most important cues, that have been discussed, are:

- holistic scene classification based on 2D as well as 3D features,
- relations between image blobs,
- co-occurrence of image features and object types,
- priming factors, focus of attention, and scale selection based on contextual image features,
- local 3D geometries,
- occlusions,
- 2D-3D aspect relationships, and
- verbal object descriptions.

The techniques discussed above utilize different representations that abstract from different scene aspects in order to become applicable. This makes it difficult to unify all aspects in a single reasoning framework. Perhaps this should even not be our goal. Rather than defining a single processing step, scene perception might be modelled as an incremental distributed process operating on a single – but multi-layer – memory structure. This partly resembles ideas from situation models and will be further discussed in Chap. 6 for the *Active Memory* approach.

Chapter 3

Perception of Scene Dynamics

In this chapter, I extend the discussion of how to integrate context into computer vision techniques towards the dimension of time. As we are looking at situated computer vision in human-machine interaction scenarios, I concentrate on scene dynamics that are caused by a human. Especially, the focus is on manipulative actions of humans. These provide a very interesting field for modeling context because pure trajectory information of the human body is not sufficient for interpretation. The entities manipulated need to be taken into account. Furthermore, manipulative actions indirectly serve a communicative goal because they give hints on human intentions and allow the artificial system to anticipate human acting.

There are two different perspectives on the modeling of human acting. Either one concentrates on the human body motion and considers other scene entities as related context. Or the representation centers on the scene entities and human body motions are used for introducing contextual relations between scene entities. In the first case, *motion models* are utilized for recognition. In the second case, *event models* are matched to the environmental state changes observed over time.

3.1 What is an action?

The scene dynamics describes how a scene evolves in time. This is an inherently continuous process. The same applies to human behavior. We are continuously moving and controlling our body. Nevertheless, everybody is able to distinctively name discrete movements, like "turning the head" or

"taking the cup". This suggests that performing and representing actions is more than a motor control problem. However, it is not entirely clear when such a named movement precisely starts, when it ends, and how its representation is internally structured. Psychologists like Bernstein (1967) and Hoffmann & Zießler (1986) at least suggest that concepts play an important role in cognitive movement representation. Schack & Mechsner (2006) even propose that there is a level of basic action concepts (BACs) similar to the basic level of Rosch & Mervis (1975) for object categories. BACs are functionally relevant elementary components of complex movements and can be characterized by recognizable perceptual features. Schack & Mechsner (2006) assume that voluntary movements are stored in and accessed from memory through their *anticipated characteristic (e.g. sensory) effect*. Examples are "turning the head" or "bending the knees". Consequently, the motor control and perception of body movements is directly linked.

In the following, we will concentrate on the recognition of human actions. Bobick (1998) introduces a taxonomy of *movements, activities*, and *actions*. While movements might roughly relate to BACs, activities refer to more complex structures like "dancing". Actions cannot be described without considering the environment as these are directed to environmental state changes. This directly relates the recognition of actions with the perception of scenes which provide contextual information.

3.1.1 Using context in action recognition

The interpretation of scene dynamics is a wide field that reaches from the analysis of the optical image flow, estimation of camera movements, tracking of objects and their articulations, recognition of motion patterns and structured sequences, the analysis of actions and state changes, maintaining an evolving scene model or an agent's situation model, up to the generation of natural language descriptions for longer episodes. Context plays an important role regarding all different levels with an increasing scope in space and time as one goes up through the different abstraction levels.

In this section, I will focus on the intermediate level of action recognition. Here, *actions* will be treated as an organized activity to accomplish an objective related to the environmental state. These can typically be referred to by verbs, like "put", "take", "pour", "shake", etc. The abstraction level roughly compares to that of object recognition with a similar variety of representational approaches. These can be divided into those that use motion

models and those that abstract from motion information,

Motion models: Motion models interpret a sequence of feature vectors $f^{(1)}, \ldots, f^{(M')}$ defined over the image sequence $v^{(1)}, \ldots, v^{(M)1}$. Typically, subsequent feature vectors are highly correlated as they do not change very much from frame to frame. Because of this kind of smoothness, we will frequently talk about trajectories in the high-dimensional feature space rather than sequences. Motion models characterize trajectories in the feature space. Typical representatives are template-based or probabilistic approaches. The former records a template trajectory and computes a distance measure between the image trajectory and the model template. During this process different transformations may be applied to the template. The latter computes a model likelihood given the image trajectory.

Event models: Event models analyze discrete state changes in the scene and deduce the actions that have lead to the state changes. Thus, actions are inherently modeled by contextual factors that define their pre- and post-conditions rather than the movement in itself. While event models abstract from the detailed realization of a trajectory, they frequently use discontinuities of trajectories for detecting events, e.g. "ball touches ground".

3.2 Action as a symbolic sequence of state-changes

Event models are mostly based on a first detection stage that transforms the image sequence into a sequence of discrete event symbols. These can be formulated as predicates in some logical formalism. However, classical predicate logic (PL-1) is not sufficient because it needs to reason about time intervals that could possibly overlap. Therefore, different kinds of extensions have been proposed.

[1] In most cases we have $M' = M$ (one feature vector per frame). However, sometimes it is more appropriate to summarize multiple frames by one feature vector or to use only selective key frames.

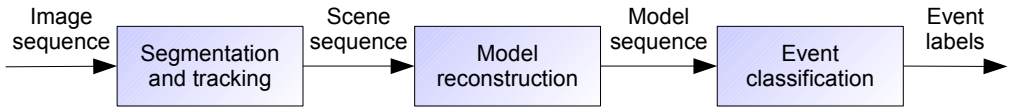


Figure 3.1: The overall architecture of LEONARD

3.2.1 Event logic

Siskind (2001) presents the LEONARD system that uses event logic in order to classify simple action verbs like "take", "put", or "move". The processing stages are shown in Fig. 3.1. The *scene sequence* consists of a set $\mathcal{P} = \{p_1, \ldots, p_{|\mathcal{P}|}\}$ of convex polygons for each frame denoting the relevant objects of a scene. Polygons are consistently ordered so that the polygon p_i corresponds to the same object in different frames. The *model reconstruction* process instantiates a set of predicates for each scene that provides a first image interpretation I based on simple force-dynamic relations,

GROUNDED(p): Polygon p is fixed in position and orientation by an unseen mechanism.

RIGID(p, q, r): Polygons p and q are attached to each other in point r by a rigid joint (relative position and angle is constrained).

REVOLUTE(p, q, r): Polygons p and q are attached by a revolute joint in point r (relative position is constrained, angle is free).

SAMELAYER(p, q) Polygons p and q are on the same depth layer and, therefore, cannot inter-penetrate.

An interpretation I can be checked by a stability analysis. Thus, a predicate STABLE($\mathcal{P}, I$) is true if the scene $\mathcal{P}$ is stable under interpretation I.

The event classification approach starts from a *model sequence*. Event logic is defined on time intervals, e.g. the event occurrence formula

$$\text{SUPPORTS}(\textbf{green-block},\textbf{red-block})@[0, 13)$$

could be true for the interval from frame 0 to frame 13. Using event logic we are able to construct compound event types out of primitive event types. As a basic notion $\Phi@\mathbf{i}$ holds if event type Φ would exactly coincidence with interval $\mathbf{i}$ (same start and end times). Compound event types can have an

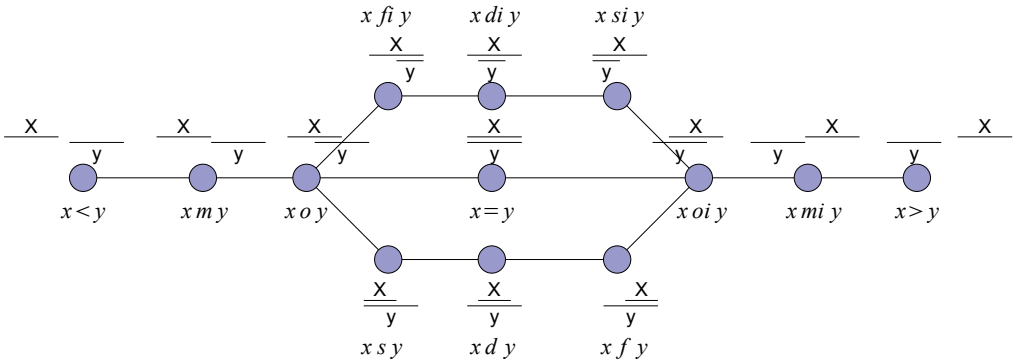


Figure 3.2: Relations between intervals: m, o, s, f, d are standing for *meet, overlap, starts, finish, and during*; mi, oi, si, fi, and di are the inverse relations (Allen, 1983).

internal temporal structure that is specified through the set of 13 possible *Allen relations* (Allen, 1983) (Fig. 3.2).

Formally, an event-logic expression is built out of *constants*, *atoms*, and *formulas*:

- *constants* denote scene objects, e.g. **red-block**,
- *atoms* are primitive event-type symbols that define n-ary relations on tuples of n constants, e.g. SUPPORT(**hand, red-block**),
- *formulas* or event-logic expressions are either atoms or combine other formulas Φ, Ψ by the following operations:

$$\neg\Phi, \quad \Phi \vee \Psi, \quad \Phi \wedge_R \Psi, \text{ or } \quad \diamond_R \Phi$$

with $R \subseteq \{=, <, >, m, mi, o, oi, s, si, f, fi, d, di\}$ a subset of the Allen relations. For example $\Phi \wedge_{\{m\}} \Psi$ denotes a compound event-type where the two event types Φ, Ψ immediately follow each other. $\diamond_R \Phi @ \mathbf{i}$ is a quantor that can act as a tense operator. It denotes an occurrence of Φ at some other time interval $\mathbf{j}$ such that $\mathbf{j} r \mathbf{i}$ for some $r \in R$, e.g. $\diamond_{\{<\}} \Phi$ means that Φ happened somewhere in the noncontiguous past.

The semantics of event logic are defined relative to a model $\mathcal{M}$ that is a map from primitive event-type symbols P of arity n to the Cartesian product of

$$
\begin{aligned}
x = y &\triangleq \overline{x = y} \\
\textsc{Supported}(x) &\triangleq \overline{\neg\textsc{Grounded}(x)} \\
\textsc{Contacts}(x, y) &\triangleq \overline{\textsc{Touches}(x, y) \wedge \textsc{SameLayer}(x, y)} \\
\textsc{Attached}(x, y) &\triangleq \overline{(\exists r)\textsc{Rigid}(x, y, r) \vee \textsc{Revolute}(x, y, r)} \\
\textsc{RigAtt}(x, y) &\triangleq (\exists r)\mathrm{Rigid}(x, y, r) \\
\textsc{Supports}(x, y) &\triangleq \overline{\neg\textsc{Grounded}(y) \wedge \neg\textsc{Stable}(\mathcal{P}\backslash\{x\}, I')}
\end{aligned}
$$

where I' is the scene interpretation I with all scene objects added to GROUNDED that are not rigidly attached to y, GROUNDED'=GROUNDED $\cup \{z|\neg\textsc{RigAtt}(z, y)\}$.

Figure 3.3: Primitive event-types used by the LEONARD system. The overline $\Phi = \overline{\phi}$ translates the PL-1 ϕ expressions to event-types Φ. x and y are variables for object constants.

the set of intervals $\mathcal{I}$ and multiple scene objects $\mathcal{O}$,

$$\mathcal{M}(P) \subseteq \mathcal{I} \times \underbrace{\mathcal{O} \times \cdots \times \mathcal{O}}_{n} \tag{3.1}$$

- $\mathcal{M} \vDash P(c_1, \ldots, c_n)@\mathbf{i}$ if and only if $(\mathbf{i}, c_1, \ldots, c_n) \in \mathcal{M}(P)$
- $\mathcal{M} \vDash (\neg\Phi)@\mathbf{i}$ if and only if $\mathcal{M} \nvDash \Phi@\mathbf{i}$
- $\mathcal{M} \vDash (\Phi \vee \Psi)@\mathbf{i}$ if and only if $\mathcal{M} \vDash \Phi@\mathbf{i}$ or $\mathcal{M} \vDash \Psi@\mathbf{i}$
- $\mathcal{M} \vDash (\Phi \wedge_R \Psi)@\mathbf{i}$ if and only if there exist two intervals $\mathbf{j}$ and $\mathbf{k}$ inside a minimal interval $\mathbf{i}$ such that $\mathbf{j}r\mathbf{k}$ for some $r \in R, \mathcal{M} \vDash \Phi@\mathbf{j}$, and $\mathcal{M} \vDash \Psi@\mathbf{k}$
- $\mathcal{M} \vDash (\diamond_R\Phi)@\mathbf{i}$ if and only if there exists some interval $\mathbf{j}$ such that $\mathbf{j}r\mathbf{i}$ for some $r \in R, \mathcal{M} \vDash \Phi@\mathbf{j}$.

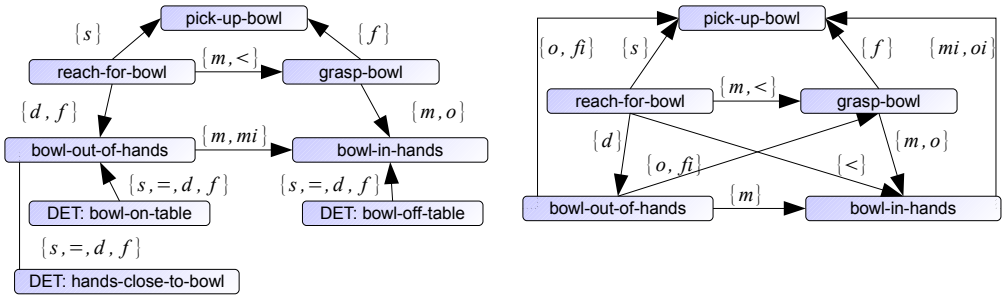
The primitive event-types are formulated in terms of the predicates GROUNDED, RIGID, REVOLUTE,SAMELAYER as well as the additional predicates $\textsc{Stable}(\mathcal{P}, I)$ and $\textsc{Touches}(p, q)$ that are defined between two polygons. Examples of primitive event-types are given in Fig. 3.3.

Then, compound event-types can be formulated as event-logic expressions. For example, the action of *picking-up* an object y with hand x from its place on z can be formulated as a compound of three related intervals (free hand x, x grasps y, x holds y)

$$\begin{aligned}
\textsc{PickUp}(x,y,z) \triangleq\ & \neg \diamond x = y \wedge \neg \diamond z = x \wedge \neg \diamond z = y \\
& \wedge \textsc{Supported}(y) \wedge \neg \diamond \textsc{Attached}(x,z) \\
& \wedge (\Phi_1 \wedge_{\{m\}} \Phi_2 \wedge_{\{m\}} \Phi_3) \\
\text{where } \Phi_1 \triangleq\ & \neg \diamond \textsc{Attached}(x,y) \wedge \neg \diamond \textsc{Supports}(x,y) \\
& \wedge \diamond \textsc{Supports}(z,y) \wedge \neg \diamond \textsc{Supported}(x) \\
& \wedge \neg \diamond \textsc{Attached}(y,z) \wedge \neg \diamond \textsc{Supports}(y,x) \\
& \wedge \neg \diamond \textsc{Supports}(y,z) \wedge \neg \diamond \textsc{Supports}(x,z) \\
& \wedge \neg \diamond \textsc{Supports}(z,x) \\
\Phi_2 \triangleq\ & \textsc{Attached}(x,y) \vee \textsc{Attached}(y,z) \\
\Phi_3 \triangleq\ & \diamond\ \textsc{Attached}(x,y) \wedge \diamond \textsc{Supports}(x,y) \\
& \wedge \diamond \textsc{Supports}(z,y) \wedge \neg \diamond \textsc{Supported}(x) \\
& \wedge \neg \diamond \textsc{Attached}(y,z) \wedge \neg \diamond \textsc{Supports}(y,x) \\
& \wedge \neg \diamond \textsc{Supports}(y,z) \wedge \neg \diamond \textsc{Supports}(x,z) \\
& \wedge \neg \diamond \textsc{Supports}(z,x)
\end{aligned} \tag{3.2}$$

The task of the event-classification component is to infer all occurrences of a given set of compound event types from a given set of primitive event occurrences. Thus, we need to find the set of intervals $\mathcal{E}(\mathcal{M},\Phi) \triangleq \{\mathbf{i}|\mathcal{M} \vDash \Phi@\mathbf{i}\}$ for all instantiations of event-type Φ with regard to the given object tuples. The computation of $\mathcal{E}(\mathcal{M},\Phi)$ is complicated by the fact that an event-type that is true for an interval is also true for all of its sub-intervals. Furthermore, $\diamond_R$ quantifies over sub-intervals and $\wedge_R$ over pairs of intervals. In Siskind (2001) a representation structure, called *spanning intervals*, and an associated inference procedure is presented that efficiently represents and operates on infinite sets of sub-intervals.

The LEONARD system has been successfully demonstrated in a simple block scenario, where a single hand is picking-up, putting-down, moving, and stapling colored blocks.



(a) IA-network (b) path consistency

Figure 3.4: Interval algebra constraint network for a "pick-up bowl" action. The prefix DET denotes relational events detected in the scene. The network in (b) has been generated out of (a) by Allen's path-consistency algorithm (intervals of detectors have been left out).

3.2.2 Constraint networks

Another example for event-based action recognition is the (past,now,future) network (PNF-network) proposed by Pinhanez & Bobick (1998). Here, the problem is formulated as a constraint network. Based on an interval algebra, they represent the inherent temporal structure of human actions.

Interval algebra networks

For this purpose, they start by modeling an *interval algebra constraint network* (IA-network) (Allen, 1983). In this model, the variables range over intervals and the arcs refer to binary temporal constraints between the intervals. Again the Allen relations (Fig. 3.2) are used for defining the temporal constraints. Fig. 3.4(a) shows an example for a "pick-up bowl" action. The network models an action including its sub-actions and object relationships detected in the scene. For a particular problem, additional unary constraints are typically assigned to some intervals fixing the variable's value. Then, a solution of the IA-network is given by an assignment of time intervals to all network variables $x_1, \ldots, x_n$ that fulfills both the binary and unary constraints. The set of all possible solutions of a variable x_i is called *minimal domain* of x_i. The solution of the detection problem, i.e. determining the minimal domains for all network variables, is NP-hard. Therefore, Pinharez and Bobick map the original IA-network to a simpler PNF-network that

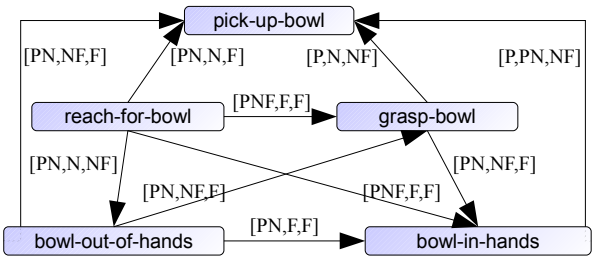


Figure 3.5: PNF-network for the "pick-up bowl" action. The network has been generated out of the IA-network shown in 3.4(b).

can be solved more efficiently and provides an approximate solution for the original detection problem.

PNF-networks

A PNF-network is a binary constraint satisfaction network where each variable $w_i, i = 1 \ldots n$ can be assigned to three different values $w_i \in M, M = \{\text{past,now,future}\}$. The possible constraints are given by the powerset of M abbreviated by

$$\mathcal{M} = \{\text{EMP,P,N,F,PN,PF,NF,PNF}\} \tag{3.3}$$

where EMP is the empty set, P stands for 'past', N for 'now', and F for 'future'. PNF specifies the complete set M.

A binary constraint W_{ij} is a truth matrix of compatible values of w_i and w_j. It can be specified by a triple $[\tau_P, \tau_N, \tau_F] \in \mathcal{M} \times \mathcal{M} \times \mathcal{M}$ with τ_P providing the constraints on w_j given $w_i = \text{past}$.

The PNF-network can be generated from an IA-network in a straight forward manner. Each node in the IA-network defines a node in the PNF-network. Each binary IA-constraint is transformed into a binary PNF-constraint by abstracting the Allen relation, i.e. they are substituted by one of the 'past', 'now', 'future' states so that the temporal order is preserved. An example is given in Fig. 3.5. The detection problem on the PNF-network is solved by an arc-consistency algorithm proposed by Mackworth (1977).

The sensor information is considered by unary constraints on the variables. If a primitive event has been detected the corresponding variable is set to the state 'now'. Past sensor information is used through a temporal

propagation method that takes the minimal domains at time step $t-1$ and uses the propagated values as unary constraints at time step t. The updated unary constraints are defined as follows:

$$W^t \leftarrow \mathcal{R}(\mathcal{T}(W^{t-1}) \cap S^t) \tag{3.4}$$

where W^t is the solution of the detection problem at time step t, S^t is the sensor information, $\mathcal{T}(.)$ is the time propagation function, and $\mathcal{R}$ computes the detection problem.

The propagation function adapts the minimal domain $\Phi \in \mathcal{M}$ of each variable to the next time step,

$$\mathcal{T}(\Phi) = \bigcup_{\phi \in \Phi} \mathcal{T}_M(\phi), \text{ with } \mathcal{T}_M(\text{past}) = \text{P}, \ \mathcal{T}_M(\text{now}) = \text{PN}, \ \mathcal{T}_M(\text{future}) = \text{NF} \tag{3.5}$$

A single evaluation of the PNF-network gives an interpretation for the current time step. Actions are recognized if 'now' is element of the minimal domain of the corresponding network variable. In order to provide the interpretation of a whole sequence, the results need to be propagated through every time step.

The approach has been successfully tested on three different actions "pick-up bowl", "wrapping chicken", and "mixing ingredients" on a video from a TV cooking show. However, the sensor values have been manually labeled.

3.3 Action as a stochastic process

Objects, actions, and scenes are tightly related. I already have discussed how the relation of objects and scenes can be modeled. Introducing actions extends the previous models by the dimension of time. Time is introduced by scene entities that are moving, this can be a moving hand or a moving object. In terms of context, we can take two different perspectives. Either the trajectories of the moving scene entities define the relevant parts of the scene, or the manipulated scene objects define the relevant parts of the trajectory.

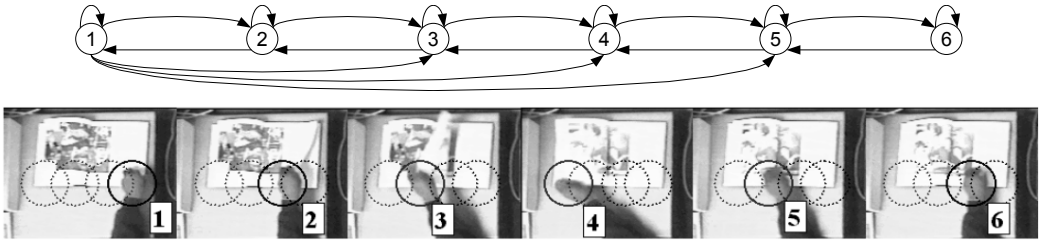


Figure 3.6: Example of a semi-ergodic HMM with 6 states corresponding to action stages of a "flip-forward" action in the context of a book. In this simple model the action stages directly refer to spatial positions of the hand. The solid circle visualizes the Gaussian activated by the activated hidden state. Example was taken from Moore (2000).

3.3.1 Probabilistic motion models

HMM-based action recognition

A Hidden Markov Model (HMM) λ models the joint probability distribution over a sequence of feature vectors $Pr(y_1, \ldots, y_T|\lambda)$. Each action $j = 1, \ldots, J$ is represented by a separate HMM model λ_j. Thus, the classification problem is given by

$$\textsc{Classify}(y_1 \ldots y_T) \triangleq \underset{j}{\operatorname{argmax}}\, Pr(y_1 \ldots y_T|\lambda_j) \tag{3.6}$$

Hidden variables $X_t, t = 1 \ldots T$ are discrete while observation variables $Y_t, t = 1 \ldots T$ may have discrete or continuous domains depending on the features. As an example for a simple choice of features, the trajectory of a hand can be described by the two-dimensional coordinates of the hand centroid (x, y) given by a color-blob tracker. The states X_t typically represent characteristic stages of an action sequence. In this example, they intuitively quantize the positions passed by the trajectory. The conditional observation density $Pr(y|x)$ can be modeled by a single Gaussian distribution with a state-specific mean and variance learned from training data (Fig. 3.6).

An HMM computes a posterior for a complete observation sequence. Thus, the input sequence is assumed to be pre-segmented into coherent meaningful sections. If we want to relax these constraints, compound models can be used in a similar way as in continuous speech recognition where it already has been explored for a long time (O'Shaughnessy, 2000). Here, a dedicated state of the HMM is marked as an ending node and the system automatically

searches over possible starting and ending times of HMM-hypotheses. It outputs the posterior of the most likely sequence of the HMM-models previously given.

State Space Models

State Space Models (SSM) have a similar structure like HMMs, but here the hidden variables $X_1, \ldots, X_T$ have a continuous domain. Kalman filters (Kalman, 1960)[2] are typical representatives that assume a Gaussian distribution that is propagated through the state sequence while processing the observations,

$$Pr(x_t|y_1, \ldots, y_{t-1}) = \mathcal{N}_{x_t}(\mu_{y_1,\ldots,y_t}, \mathbf{K}_{y_1,\ldots,y_t}), t = 1, \ldots, T \tag{3.7}$$

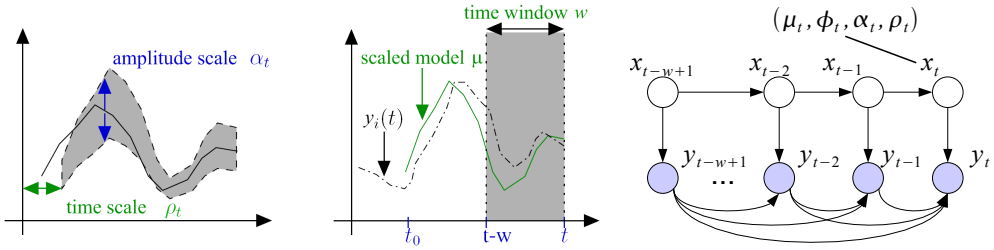
Approaches that loosen assumptions with regard to the class of distributions need approximate inference algorithms in order to be tractable. A successfully applied technique is the CONDENSATION (CONDitional dENSity propagATION) algorithm (Isard & Blake, 1996, 1998) that is based on a variant of sequential importance sampling (Arulampalam *et al.*, 2002). Isard and Blake apply it to tracking tasks and Black & Jepson (1998) transfer the framework to the matching of template trajectories. This would correspond to a pure motion model which is not sufficient for action recognition. Therefore, Fritsch *et al.* (2004) and Hofemann *et al.* (2004) add symbolic context information to the framework.

The model base is given by a set of M trajectories $\{\mathbf{m}^{(\mu)}, \mu = 1, \ldots, M\}$ that each represents a template for a different type of movement. The models are defined as discretely sampled, interpolated curves with a phase parameter ϕ and an N-dimensional vector $\mathbf{m}_\phi^{(\mu)} = (m_{\phi,1}^{(\mu)}, \ldots, m_{\phi,N}^{(\mu)})$ describing the model values at position ϕ. The observations are given by an input trajectory that is also represented by a sequence of N-dimensional vectors $\mathbf{y}_t = (y_{t,1}, \ldots, y_{t,N})$. The internal state of the probabilistic model is given by a set of matching parameters at time t:

$$\mathbf{x}_t = (\mu_t, \phi_t, \alpha_t, \rho_t) \tag{3.8}$$

where μ_t is an integer indicating the model matched at time t, ϕ_t is the position or phase within the model that is aligned, α_t is an amplitude parameter for scaling the model values at time t, ρ_t is a parameter for scaling the model in the time dimension.

[2]Kalman filters are typically applied for tracking application, e.g. Wren *et al.* (1997)



(a) scaling parameters (b) model matching (c) graphical model

Figure 3.7: Trajectory matching of model μ on time window $(t-w)\dots t$. The state parameters α, ρ are fitting the model trajectory (green) to the observation $y_i(t'), t' = (t-w)\dots t$. Figures (a) and (b) have been taken from Hofemann (2007).

The observation model is defined over a sliding temporal window with size w (Fig. 3.7). The components of an observation model are assumed to be conditionally independent

$$Pr(\mathbf{y}_t|\mathbf{y}_{t-1},\dots,\mathbf{y}_{t-w+1},\mathbf{x}_t) = \prod_{i=1}^{N} Pr(y_{t,i}|y_{t-1,i},\dots,y_{t-w+1,i}|\mathbf{x}_t)$$

where

$$Pr(y_{t,i}|y_{t-1,i},\dots,y_{t-w+1,i},\mathbf{x}_t) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp \frac{-\sum_{j=0}^{w-1}(y_{(t-j),i} - \alpha_t m^{(\mu)}_{(\phi-\rho j),i})^2}{2\sigma_i(w-1)} \tag{3.9}$$

The dynamics of the model $Pr(\mathbf{x}_t|\mathbf{x}_{t-1})$ is defined as follows. Given a state $\mathbf{x}_{t-1}$ the variable $\mathbf{x}_t$ is distributed as

$$\mu_t = \mu_{t-1}; \qquad \phi_t = \phi_{t-1} + \rho + \mathcal{N}(\sigma_\phi); \tag{3.10}$$

$$\alpha_t = \alpha_{t-1} + \mathcal{N}(\sigma_\alpha); \qquad \rho_t = \rho_{t-1} + \mathcal{N}(\sigma_\rho) \tag{3.11}$$

The model state $\mathbf{x}_0 = (\mu_0, \phi_0, \alpha_0, \rho_0)$ is initialized by random sampling in a range of possible values,

$$\mu_0 \in [1, M]; \qquad \phi_0 = (1-\sqrt{z})/\sqrt{z} \text{ with } z \in [0,1]; \tag{3.12}$$

$$\alpha_0 \in [\alpha_{min}, \alpha_{max}]; \qquad \rho \in [\rho_{min}, \rho_{max}]. \tag{3.13}$$

Each sample defines a particle that is propagated through the model. Thus, the state posterior in each time step is represented by the weighted set of

particles. Given the pure dynamics of the probabilistic model, a single action is recognized by matching a set of trajectory templates to an input trajectory (the action class μ does not change over time and ϕ only progresses through the model trajectories). However due to the re-sampling technique applied in the CONDENSATION algorithm, a fraction of $5 - 10\%$ of the particles in each time step is exchanged by random initial guesses. Thus, a part of the matches under consideration is continuously re-initialized. Thereby, multiple matches and a spotting of actions in longer unstructured trajectories become possible.

Because the action model index is coded into the state space model, it provides an estimation of the posterior of the action classes for each time step. Therefore, the current particle states are summarized computing the marginal over the state posterior conditioned on a near maximum value of ϕ,

$$Pr(\mu|\mathbf{y}_{1:t}, \phi + 1 > \phi_{max}) = \sum_{n=1}^{S} \begin{cases} \pi_t^{(n)} & \text{if } \mu_t^{(n)} = \mu \wedge \phi_t^{(n)} + 1 > \phi_{max} \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

where $(\mathbf{s}_t^{(1)}, \ldots, \mathbf{s}_t^{(S)})$ is the set of particles with weights $(\pi_t^{(n)})_{n=1\ldots S}$ and state $(\mu_t^{(n)}, \phi_t^{(n)}, \alpha_t^{(n)}, \rho_t^{(n)})_{n=1\ldots S}$.

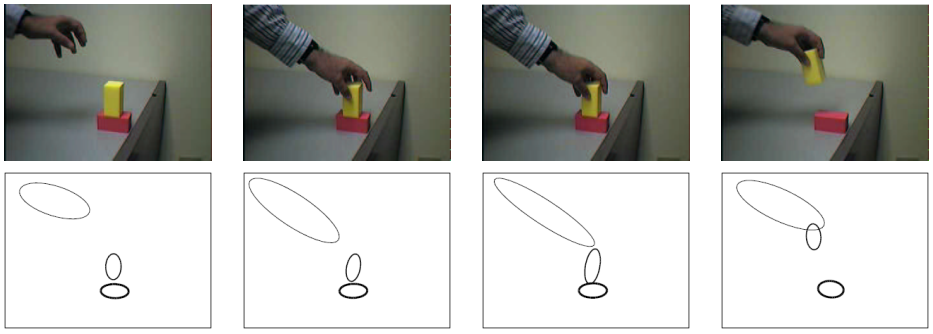
They are called *end probabilities* of action models. Actions that occur in a time stream observed can be detected by thresholding these end probabilities.

3.3.2 Using context in motion models

So far, both approaches discussed are based on pure trajectory information for action classification. There are two different principled ways of including contextual information in the HMM and SSM frameworks. Either one extends the observed feature vectors by contextual cues providing more diagnostic support, or the state representation is extended for coding aspects of the scene. This would provide additional causal support.

Coding context in feature vectors

Siskind & Morris (1996) recognize simple motion events like 'pick-up', 'put-down', 'push', 'pull', 'drop', and 'throw' using non-ergodic HMMs (Fig. 3.8).



Frame 0 Frame 24 Frame 29 Frame 44

Figure 3.8: Sample frames from a 'pick-up'-action. Each object that is described by an ellipse takes a specific role during action performance. These roles are characterized by absolute as well as relative features. Taken from Siskind & Morris (1996)

The feature vectors are defined on a collection of relevant objects that have certain roles with regard to the event class. Each object is characterized by an ellipse. On this collection, they compute absolute as well as relative features:

absolute features for each ellipse: (1) velocity of the center, (2) velocity orientation of the center, (3) angular velocity, (4) first derivative of the area, (5) first derivative of the eccentricity, (6) first derivatives of the above 5 features,

relative features for every pair of ellipses: (1) distance between the centers, (2) orientation of the vector between the centers, (3) angle between both major axes, (4) angle between major axes and the vector between the centers, (5) first derivative of the above 4 feature.

A critical aspect of this approach is the correspondence problem, i.e. one has to assign model roles to the image ellipses detected. This is complicated by possible occlusions that occur during action performance as well as additional background ellipses that do not have a role in the model. These two cases possibly corrupt the solution of the correspondence problem. In Siskind & Morris (1996) the correspondence problem is solved by testing all permutations against the HMM model. This is only tractable for a very small number of roles.

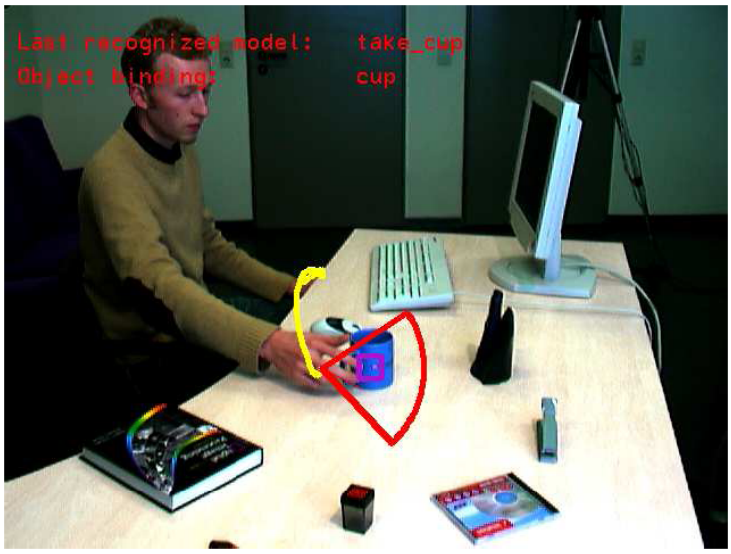


Figure 3.9: Using a context area for action recognition in an office scenario. Here the action *take-cup* has been recognized leading to a binding of *cup* for the hand context. Taken from Fritsch *et al.* (2004).

Hofemann *et al.* (2004) present a more efficient approach to the correspondence problem at the cost of a less structured incorporation of contextual information. They follow the approach of the state space model described above and use the CONDENSATION algorithm for inference. Besides a hand trajectory template, a model consists of a sequence of contextual areas $(\mathbf{c}_t)_{t=0\ldots\phi_{end}}$ including a symbolic object type context c_{type}. The contextual areas $A_t(\phi_t) = \text{CIRCLESEGMENT}_{\phi_t}(c_{orient,t}, c_{\alpha,t}, c_{\beta,t}, c_{r,t})$ are defined by a circle segment relative to the trajectory position and orientation at time t. Here, c_{orient} is the angle relative to the trajectory orientation where a context object is expected. For objects that have a specific 'handling direction' this orientation is fixed to an absolute value; c_α, c_β define the start/end angle of the circle section and c_r the radius. An additional object importance c_{imp} specifies if the presence of the object type is irrelevant, necessary, or optional. In order to include the symbolic information of the contextual area into the recognition approach, the computation of the weight update of a particle is

changed to

$$\pi_t^{(n)} \propto Pr(\mathbf{y}_t, \Theta_t|\mathbf{s}_t^{(n)}) \triangleq Pr(\mathbf{y}_t|\mathbf{s}_t^{(n)})Pr(\Theta_t|\mathbf{s}_t^{(n)}) \tag{3.15}$$

where

$$\Theta_t \triangleq \exists[O_t = (x, y, c_{type,\phi_t}) \in \mathcal{O}_t](x, y) \in A_t(\phi_t) \tag{3.16}$$

$$Pr(\Theta_t|\mathbf{s}_t^{(n)}) = \begin{cases} P_{present}, & \text{if } \Theta_t = \textsc{True} \\ P_{missing}, & \text{otherwise} \end{cases} \tag{3.17}$$

Here, $\mathcal{O}_t = \{O_{t,1}, \ldots, O_{t,K}\}$ is the scene representation with K scene objects $O_{t,k} = (x, y, o)$ described by the center (x, y) and the object class o, $A_t(\phi_t)$ is the contextual area and c_{type,ϕ_t} the expected object class of the trajectory at time t. In order to deal with possible occlusions by the hand, a global hand state is introduced that fixes the hand context after a *pick-up* action. The approach has been successfully applied to pointing gestures and typical actions in an office scenario (*take-cup, stop-drinking, pick-up-phone, hang-up-phone, pick-book, stop-reading, type-on-keyboard*) achieving recognition rates above 87%, for some actions near 100%.

Coding context in state representations

The previous approach considers a *dynamic context* that provides additional feature cues besides the positional and directional information of the trajectory. However, from a different perspective many actions can also be described as trajectories relative to a *static context* given by the scene objects. Dependent on the activated part of the scene, we can re-parameterize the observation models as well as select appropriate action models in order to facilitate action recognition.

For this purpose Moore *et al.* (1999) introduce an object-oriented framework, called *Object Spaces*. Moore refers to object-orientation with regard to both the software design philosophy – in terms of modularity, inheritance hierarchies, and the coupling of information and behavior – and the knowledge representation and control structures used for recognition purposes (Moore, 2000). Here we will concentrate on the second aspect.

In their framework, Moore et al. distinguish three different representational layers (Fig. 3.10),

- The *extraction layer* is responsible for low level tracking of hands and scene objects – called articles. HMM models are used for classifying manipulative actions from hand trajectories.

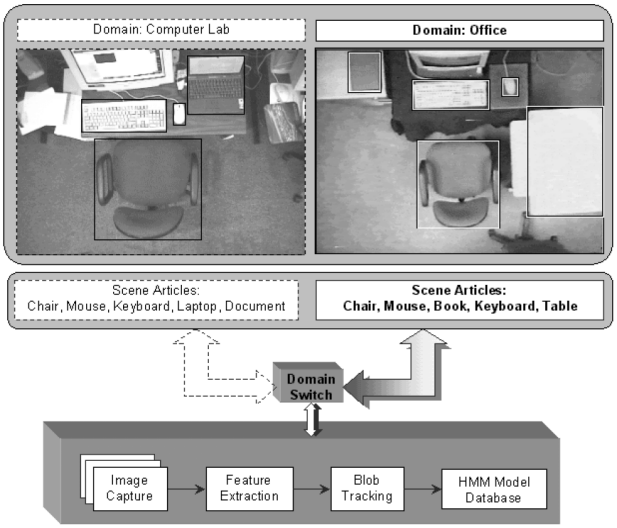


Figure 3.10: Object spaces: object-oriented constructs are exchanged between different adjacent layers. Domains can be easily switched by reusing object definitions. Image taken from Moore (2000).

- The *object layer* provides a repository for representing people and articles in the scene. Each article is characterized by a bounding box defining its relevant area in the scene.

- The *Scene layer* contains domain specific contextual models and monitors physical contact relations between people and articles.

Context is modeled as a bottom-up and top-down interactions of these representation layers. Articles are organized in a hierarchy of *generalized class models* (GCMs) containing region- and image-based descriptions. Furthermore, each class has an associated set of compatible actions. The graphical model in Fig. 3.11 shows the principled way how to incorporate a scene state into an HMM-based action recognition approach. Instead of adding contextual cues, the relation between the hidden state variables $(x_t)_{t=1,\ldots,T}$ and the observation variables $(y_t)_{t=1,\ldots,T}$ is influenced by further scene evidence. In Moore et al., each scene object O_k has an article bounding box $\mathbf{z} = (x_l, y_l, x_r, y_u)$ that defines an area of relevant motions and a local coordinate system for normalizing hand positions $\mathbf{Y} = [y_1, y_2, \ldots, y_T]$,

$$\hat{\mathbf{Y}} = \mathbf{S}_\mathbf{z}[\mathbf{R}_\mathbf{z}(\theta)\mathbf{Y} + \mathbf{y}_{trans}] \tag{3.18}$$

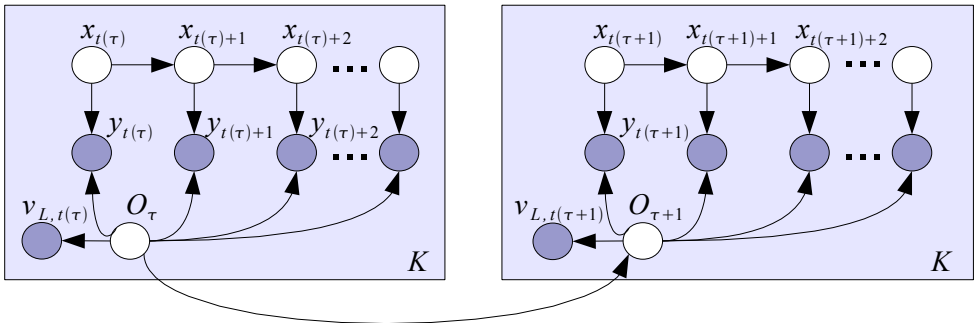


Figure 3.11: Graphical model for incorporating scene context in the hidden state of an HMM-based action recognition approach. The variables $O_\tau^{(k)}$ encode the class and bounding box of the k-th scene object at time interval τ. This information is extracted from the image information $v_L^{(k)}$. During the trajectory section $t(\tau) \dots t(\tau) + T_\tau$ the hand stays in contact relation to object O_k. The contact relation changes from time interval τ to $\tau + 1$.

where $\mathbf{S}$ is a diagonal scaling matrix with factors $S_{xx} = \mu_{width}/(x_r - x_l)$ and $S_{yy} = \mu_{height}/(y_l - y_u)$, $\mathbf{R}(\theta)$ is a rotation matrix with angle θ. Both are parameterized by the bounding box $\mathbf{z}$. $\mathbf{y}_{trans}$ is a translation vector $(-x_l, -y_u)$, and μ_{width}, μ_{height} are the mean width and height of the GCM bounding box.

Contacts between hands and articles are detected at the scene layer and imply a change of the local coordinate system. Each scene object has access to a finite state machine that changes from *inactive* to *tentative* with the initial contact of a hand region and from *tentative* to *active* in case of a non-transitive contact relation. The positional information of the hand trajectory is buffered and passed to the extraction layer for matching the hand trajectory to pre-trained HMM models related to the article. A possible displacement of the article during the interaction is stored as an update of the bounding box.

The model shown in Fig. 3.11 can also be used for incorporating action context to object recognition. Querying the variable $O_{\tau+1}$ fuses evidence from the image $v_{L,t(\tau+1)}$, the hand trajectory $(y_{t(\tau+1)}, \dots, y_{t(\tau+1)+T_{\tau+1}})$, and the object from the previous hand contact O_τ. This has been formulated by Moore et al. as a separate Bayesian network depicted in Fig. 3.12.

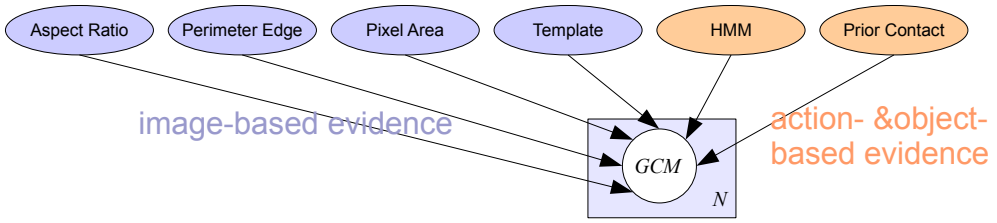


Figure 3.12: Bayesian network for relating evidence from image-based, action-based, and object-based evidences. The variable *Prior-Contact* models the object type previously activated by the hand trajectory. $GCM_{n=1...N}$ are binary variables indicating the classification of the *generalized class models*.

Extending object spaces

A related approach to the *object spaces* proposed by Moore *et al.* (1999) is put forward by Li *et al.* (2006, 2007). In their framework, the scenario constraints are somewhat relaxed. While Moore et al. assume a camera view from the top of the scene, here, camera positions are more freely chosen. Furthermore, actions take place in the vicinity of objects rather than on the object area itself. This has a couple of conceptual implications. Trajectories become view-dependent and not all parts of the trajectory inside the object vicinity provide meaningful information for an action observed. Thus, an action spotting approach is needed rather than the interpretation of a perfectly pre-segmented trajectory.

Fig. 3.13 shows the anticipated scenario. Actions are performed on a table and are observed from a pan-tilt camera on a mobile robot with a fixed height. Thus, the 2D-view can be roughly described by the distance of the robot to the table and by the pan-/tilt-angles, and the focal length of the camera. The objects $O_k = (x, y, w, h, c_{type})_k$ with $k = 1 \dots K$, that have been detected in the scene, are described by a bounding box $(x, y, w, h)_k$ and related radius r_k estimated from a SIFT-based object recognizer. A parameter β controls the relative size of the ellipse defining the object vicinities:

$$a_k = r_k \cdot \beta \cdot \cos(\arctan(o'_x/f)) \tag{3.19}$$

$$b_k = r_k \cdot \beta \cdot \sin(c_t + \arctan(o'_y/f)) \tag{3.20}$$

where a_k, b_k are the horizontal and vertical semi-axes of the ellipse, c_t is the tilt-angle of the camera, and o'_x, o'_y are the offsets of the object position $(x, y)_k$ with regard to the image center. f is the focal length

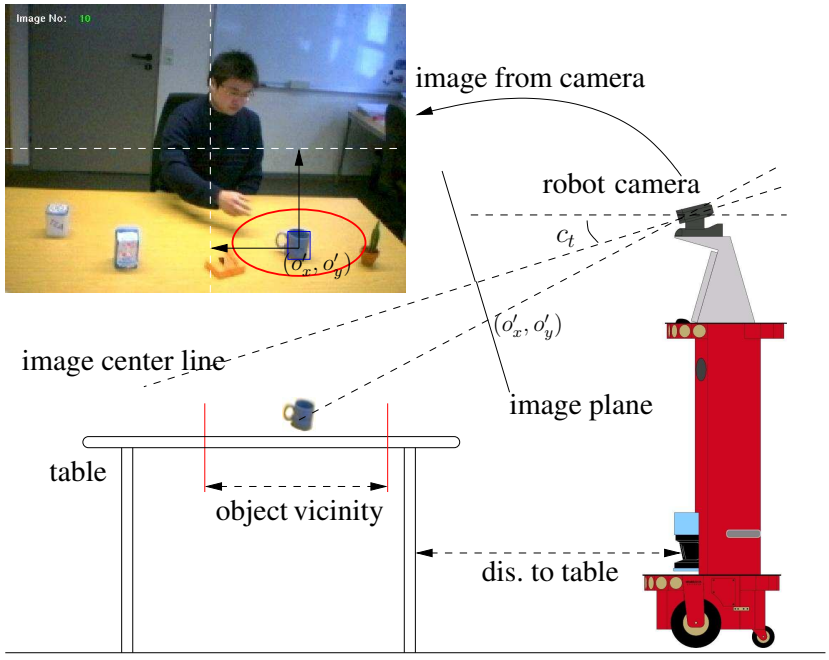


Figure 3.13: Definition of the object vicinity.

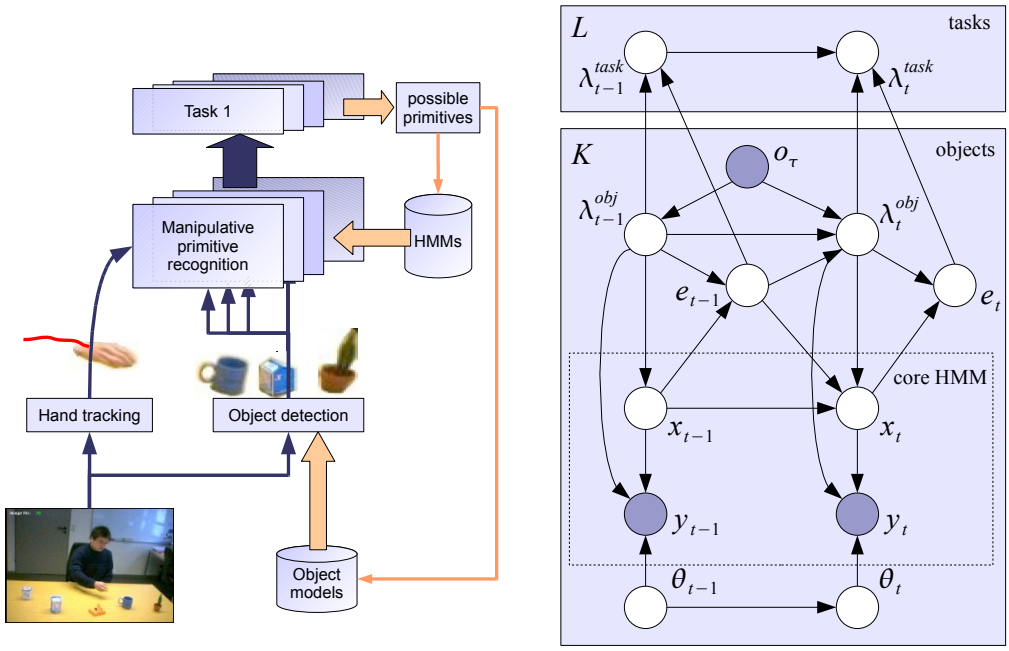
of the camera that has previously been given as an internal camera parameter.

The object vicinities are used for a coarse pre-segmentation of the hand trajectory $(h_{x,t}, h_{y,t})_{t=1...T}$ and determine a local coordinate system for computing trajectory features relative to the manipulated object. Overlapping object vicinities are processed in parallel as shown in Fig. 3.14(a). The trajectory-segments are characterized by the following features

$$\mathbf{y}_{1:T}^{(k)} = (d_t^{(k)}, v_t^{(k)}, \gamma_t^{(k)})_{t=1...T} \tag{3.21}$$

where $d_t^{(k)} = |(\frac{h_{x,t}-x_k}{a_k}, \frac{h_{y,t}-y_k}{b_k})|$ is the distance to the object center (x_k, y_k) in normalized hand coordinates (Eq. 3.18) using scaling matrix $\mathbf{S}$ with $\mathbf{S}_{xx} = 1/a_k$ and $\mathbf{S}_{yy} = 1/b_k$; $v_t^{(k)}$ is the magnitude of the hand speed; $\gamma_t^{(k)}$ is the angle between the direction of the hand trajectory and the line connecting the hand position and the object center.

The trajectory defined by the sequence of feature vectors systematically depends on the pan and tilt view angles of the camera. This is compensated



(a) System overview (b) Graphical model

Figure 3.14: Hierarchical HMM model for incorporating scene context.

by a linear scaling factor $\theta_{t,j}$ that is introduced for each component of the feature vector $y_{t,j}$. This parameter is dynamically optimized during evidence propagation.

In Fig. 3.14(b) the complete graphical model of the object-oriented approach is given. The core structure is an HMM with hidden states $\mathbf{x}_t$ and observations $\mathbf{y}_t$. This part is also trained using a classical Baum-Welch algorithm on a set of labeled trajectories segmented by hand. For reason of clarity, the depicted model abstracts from the dependencies introduced by the transformation of the trajectory coordinates into an object-specific local coordinate system. As an additional factor, we introduce hidden variables $\theta_t^{(k)}$ that adapt the observation model with regard to the actual camera view,

$$Pr(y_{t,j}^{(k)}|\mathbf{x}_t^{(k)}, \theta_{t,j}^{(k)}) = \frac{1}{\sqrt{2\pi}\sigma_{\mathbf{x}_t^{(k)},j}} \exp \frac{-(y_{t,j}^{(k)} - \theta_{t,j}^{(k)} \cdot \mu_{\mathbf{x}_t^{(k)},j})^2}{2\sigma_{\mathbf{x}_t^{(k)},j}} \tag{3.22}$$

where k indexes the scene object; j is j-th component of the observation vector $\mathbf{y}_t^{(k)} = (d, v, \gamma)_t^{(k)}$; each component $y_{t,j}^{(k)}$ is assumed to be Gaussian

distributed with $\mathcal{N}(\mu_{x,j}, \sigma_{x,j})$ where x is a discrete hidden state of the core HMM structure. The center of the Gaussians are linearly scaled by $\theta_{t,j}^{(k)}$.

In order to deal with variations in segmentation and multiple elementary actions per trajectory segment, further hierarchically organized hidden states are introduced. The $e_{t-1}^{(k)}$ variable is boolean and denotes the possible ending of an elementary action. In this case, the current core-HMM indexed by $\lambda_{t-1}^{(k)}$ might change to a new one indexed by $\lambda_t^{(k)}$. This choice is influenced by the object class context o_τ that is stable during the trajectory section of time interval τ. Furthermore in this case, the next hidden state variable x_t will be initialized by the prior state probability of the core-HMM λ_t^{obj}.

The object-specific evidence for elementary actions is collectively passed to a task-layer including L different task models. All task models are examined in parallel. Each time an ending state ($e_t^{(k)} = true$) of an elementary action λ_t^{obj} is detected, the action class is assigned to the corresponding task variable λ_t^{task}. Otherwise the task variable stays the same. Different tasks are modeled as first-order Markov chains with transition probabilities $Pr(\lambda_t^{(k)}|\lambda_{t-1}^{(k)})$ estimated from a labeled training set.

The inference problem of the graphical model is solved by de-coupling the object-action and task sub-networks. For each object k, the CONDENSATION algorithm is applied using a set of weighted particles $\{(s_t^{(n)}, w_t^{(n)})\}_{n=1...N}$ with

$$s_t^{(n)} = (\lambda_t^{(n)}, x_t^{(n)}, e_t^{(n)}, \theta_t^{(n)}), \quad w_t^{(n)} = \frac{Pr(y_t|s_t^{(n)})}{\sum_{n'=1}^{N} Pr(y_t|s_t^{(n')})} \tag{3.23}$$

The ending probability $P_{end,t}$ of an elementary action is computed by

$$P_{end,t}(\lambda^{obj}) = \sum_{n|\lambda_t^{(n)}=\lambda^{obj}} w_t^{(n)} \tag{3.24}$$

If $P_{end,t}$ succeeds a certain threshold, the evidence is considered for the task models. These can also be used in order to compute a *look-ahead* action class. Because those are defined in an object-specific manner, e.g. *take-cup*, this provides top-down information for both, the selection of relevant elementary action classes (HMMs) and the selection of scene objects relevant for the next manipulative action. Thus, it defines an attention-filter that reduces the number of parallel threads in action recognition (Fig. 3.14(a)).

3.4 Scene evolution

In the approaches discussed so far, the scene provided the context for the interpretation of the movements. However, the argumentation can also be turned around. The dynamics provide a context for the interpretation of scenes and is captured in *scene evolution models*.

Scene evolution models can be defined on different interpretation layers. A good example is given by Kolonias *et al.* (2007b,a) in the area of sports video annotation. On the lowest level, neighboring video frames of a tennis video are used to construct a global view of the tennis court. At the same time, moving foreground objects of the scene are extracted exploiting their different time dynamics. For this purpose, mosaicing techniques are applied like Shum & Szeliski (1997). The registration of each image frame is provided by a scene point matching algorithm that exploits the spatio-temporal context of scene points given by a SUSAN corner detector. Then, foreground blobs are detected by taking the differencing image between the warped test frame and the global scene view.

Players and ball can be recognized by size, shape, and time dynamics. While the players are moving rather slow, the ball is moving very fast. Kolonias *et al.* (2007b) filter false positive detections of balls by applying a second-order motion model on segments of ball flights and by combining segments to complete ball trajectories, called 'plays'.

In the tennis scenario, also high-level knowledge constrains possible interpretations because the game follows dedicated tennis rules. Kolonias *et al.* (2007b) formulate these as state models capturing possible transitions between detected events. HMMs use these state models as a hierarchical transition model in order to recognize tennis game concepts like 1st service, 2nd service, or score keeping. They show that annotation results significantly increase with considering contexts on different interpretation levels.

Similar concepts of scene evolution have also been exploited in other scenarios like structured table scenes (e.g. breakfast) (Matas *et al.*, 1998).

3.5 Summary and conclusion

The recognition of scene dynamics is an essential aspect for characterizing and maintaining situational context. In this chapter, I have concentrated on an intermediate interpretation level: manipulative action recognition. There

are many reasons why manipulative actions are a central aspect for a situated modeling of computer vision:

- They provide a natural interface for the verbalization of scene dynamics, i.e. there are many language verbs naming manipulative actions. As a consequence, this is a potential level for grounding words into perceptions;
- they inherently relate human activity to the environmental scene, i.e. both provide each other a focus and an additional meaning with regard to the current situation. Objects take roles and body movements become purposive;
- manipulative actions provide a key towards object categorization which is mostly based on function rather than appearance;
- they implicitly communicate human feedback and human intentions. If the human communication partner proceeds in performing an expected task, it is an implicit non-verbal confirmation. During a task performance a human communication partner typically assumes intention reading capabilities which simplifies interaction.

In the previous sections, I have shown different modeling techniques that incorporate context into recognition approaches. All of them have certain limitations. It is not possible to generally state beforehand, that motion models might not be relevant because manipulations aim at scene changes rather than specific hand movements. It depends on what is observable. Certain hand movements indicate scene changes and sometimes it is not observable if a piece of sugar is in the cup or not.

An essential drawback of motion models is that they are only partially decomposable and suffer from the need of large amounts of training data for capturing the variability of the relevant movements. Further research should aim at more sophisticated features and models that are able to reduce this variability. One avenue is abstraction, i.e. the definition of more invariant features. A second avenue is to model factors that are causing a systematic variability of the movement. A third avenue, is individual adaptation, i.e. person-specific models are trained that are more specific and have less variability.

In this sense, event models have a very high degree of abstraction. They perceive the scene through a set of predicates that completely abstract from

motion trajectories. The high degree of invariance is paid at the expense of model precision. Any detail that remains relevant for action recognition needs to be modeled as a discretized event. As a consequence, logic formulations are becoming bulky. They also include the risk to miss relevant environmental changes because predicates are typically decided without considering the context of the complete model.

Object recognition has shown that the most successfull approaches are based on a broad pooling of evidence. A similar approach is needed for the recognition of actions. However, actions are a more complicated domain because personnel styles as well as timing and synchonization issues come into play when considering contextual cues.

Chapter 4

Cross-situational Learning

How do we acquire and represent meaning? This is multi-faceted question that has no simple answer. It has been debated in philosophy for centuries and will be answered very differently from a psychological, neurophysiological, or computational perspective. From a technical or mathematical viewpoint the question is ill-posed. The problem is not clearly defined and a huge number of possible solutions might exist. In this chapter, I will cut down the problem to a treatable aspect of it and define meaning as an association with an already established knowledge representation. This has a couple of practical consequences. If person A can read and understand English texts, we can define the meaning of a French text for person A by translating it into English. Thus, person A is able to understand French texts if we provide him or her with tools that automatically translate French to English. Certainly, one could argue that the tool needs to understand French texts in order to *correctly* translate *arbitrary* French sentences into English sentences. However, this is not our goal! We are only looking at a *domain-specific subset* of French sentences and allow the translation model to *make errors* by taking a statistical approach.

In this chapter, I will not discuss the problem of human language translation but I will partially transfer techniques for statistical language translation to other modalities and especially use them across modalities. A tool that automatically translates visual information to sets or sequences of textual or spoken words and vice versa would open up a variety of applications.

- Automatic image annotation enables retrieval tasks in large unstructured image collections by simply specifying a set of keywords.

- Automatic text illustration provides image candidates for enriching large corpora of pure text.
- The automatic integration of otherwisely separated image and text resources could lead to a new level of automatic knowledge discovery.
- Long videos or video sections could be summarized by a few characterizing words.
- In human-machine and human-robot communication the artificial system could verbalize what it perceives. Currently, this is only the case for a few hand-selected visual events. Learning such models from larger corpora would enable the verbalization of a much broader spectrum of visual events.
- Expectations on what a human might say and what a human refers to while speaking will significantly robustify human-machine communication and interaction.

The problem of learning a *visual translation model* that translates images into words or text is similar or mostly identical to acquiring models for general computer vision tasks like object recognition, object detection, scene classification, gesture and action recognition, or image (sequence) understanding. The only difference lies in the perspective on the training data. Traditionally, visual models are learned from labeled or partially labeled training data.

Taking a typical pattern recognition approach, one either directly estimates the parameters θ_q of a conditional distribution $Pr(\mathbf{y}|q;\theta_q)$ that represents the visual model of class q. Here, a training set is needed where each feature vector $\mathbf{y}$ is labeled by its class. Or, one estimates the visual model as a mixture of N components. Although the components can be learned from a possibly unlabeled part of the training set, the mixture weights are estimated from a labeled training subset.

$$Pr(\mathbf{y}|q;\theta_q,\theta_{n=1\ldots N}) = \sum_{n=1}^{N} Pr(\mathbf{y}|Z=n;\theta_n)Pr(Z=n|q;\theta_q). \tag{4.1}$$

In both cases, one needs a subset of *strongly labeled* data items. This means that the features extracted from the training set are already grouped into meaningful collections of features that refer to class representatives and that

the mapping from image features to model features is given. In typical object recognition tasks, this is provided by the boundary or bounding box of an object. Parts of the bounding box that do not refer to a class representative are treated as random noise. The construction of strongly labeled datasets is mostly done by hand and is a huge endeavor.

In order to reduce the large amount of effort, many approaches have been proposed that require only *weakly labeled* data sets. Thus, training sets can be chosen from more unconstrained data collections. These only need to fulfill the restriction that an instance of the class referred to by a label is present somewhere in the data item. An example of such an approach is given by Carneiro *et al.* (2007) for automatic image annotation or Crandall & Huttenlocher (2006) for part-based object recognition.

The common perspective in all variants of labeled training sets is that, there is a supervised process (a human annotator) that provides (weak or strong, complete or partial) labels for a large collection of data items that have been generated by some stochastic process. The perspective taken by statistical translation models is different in that the treatment of the translation source (before data items) and translation goal (before labels) is symmetric. Both parts are assumed to be generated by related stochastic processes. Taken differently, both parts provide a *context* for each other that is the basis for learning meaningful inter-relationships. Because of this perspective I term it *parallel datasets*.

4.1 Parallel datasets

I will talk of parallel datasets if there is a resource that consists of two kinds of data and that is organized in paired data-items (one of each kind) that share a partial meaning.

Definition 1 (Parallel dataset) *A parallel dataset consists of pairs of collections* $(\mathbf{e}^{(s)}, \mathbf{f}^{(s)})_{s=1\ldots S}$ *with*

$$\mathbf{e}^{(s)} = (e_1^{(s)}, e_2^{(s)}, \ldots, e_{L^{(s)}}^{(s)}), \quad \mathbf{f}^{(s)} = (f_1^{(s)}, f_2^{(s)}, \ldots, f_{M^{(s)}}^{(s)}), \quad s = 1, \ldots, S$$

such that there is an alignment $\mathbf{a}^{(s)} = \{(\alpha_j^{(s)}, \beta_j^{(s)})\}_{j=1\ldots J}$ *that links subcollections* $\mathbf{e}^s_{\alpha_j^{(s)}} \subseteq \mathbf{e}^{(s)}$ *and* $\mathbf{f}^s_{\beta_j^{(s)}} \subseteq \mathbf{f}^{(s)}$ *with a similar meaning.*

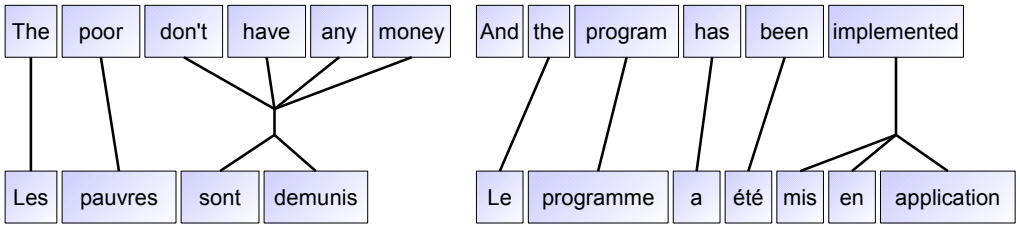


Figure 4.1: A short parallel text that consists of two paired English and French sentences. An alignment of English and French words can map single words as well as multiple words to single and multiple words. The word 'And' is mapped to the empty word.

Fig. 4.1 shows an example of an English/French parallel text. Alignments between parallel datasets can involve several groupings on both sides in order to provide meaningful mappings.

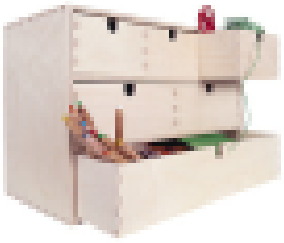
Parallel datasets are an interesting source for learning models because they offer semantic access to huge collections of unstructured data. Typically, parallel datasets have not been built up with the purpose of model acquisition in mind. As a consequence, resources are omnipresent. Large corpora of parallel texts are found in the documentation for the Canadian parliament or the EU administration. Examples of multi-modal parallel datasets are museum image collections, web archives, catalogues, collections of news photographs, private photo galleries, etc. But these kinds of data are also partially challenging in terms of appropriate features selection, feature grouping, and ambiguity (synonyms and homonyms). In Fig. 4.2 a few examples for parallel datasets are shown – in this case paired images and captions. One can see that not all words in the captions have reasonable correspondents in the images, e.g. 'blowing' or 'mini', some refer to image qualities, e.g. 'print', some are proper names, e.g. 'MOPPE', some have visual ambiguities, e.g. the 'chest' in (f) looks much more like the 'wardrobe' in (e) than the other chest in (d). Nevertheless, we can identify words in the caption that refer to specific parts of an image, e.g. 'tiger', 'boat', 'wardrobe'. The goal is to filter these relevant words and to find a consistent mapping on all paired data items.

The principles of translation models have already been explored on parallel texts (or bi-texts) for a long time in the field of statistical machine translation. Given a large corpus of parallel English-French texts that con-

(a) syntax lord print rowlandson

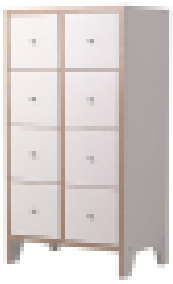
(b) print small boat approaching blowing whale ship mountain background currier

(c) tiger cat water grass



(d) MOPPE mini chest of drawers

(e) LEKSVIK wardrobe with 2 doors



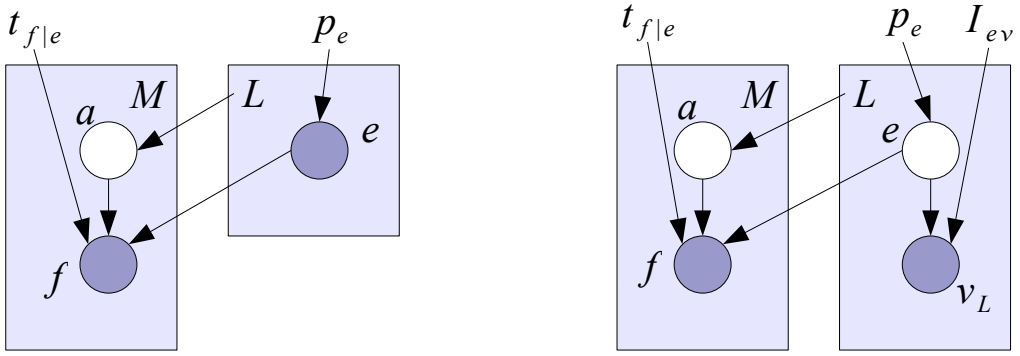
(f) ROBIN chest of drawers with 8 drawers

Figure 4.2: Parallel datasets from the Fine Arts Museum of San Francisco (a+b), the Corel data set (c), and an IKEA furniture catalog (d-f).

sists of pairwisely grouped sentences that translate into each other, they learn a translation model that is able to automatically annotate a new French sentence with an English word sequence resembling its semantic meaning. If we are able to learn a visual vocabulary, we can transfer this approach by exchanging French with visual words and English with caption words. Other paradigms explore co-occurrence statistics and try to maximize mutual information measures. Other approaches define general rules of cross-situational inference in order to map symbolic structures (Siskind, 1996).

4.2 Statistical translation models

In statistical language models, we generally seek to find the translation string $\mathbf{e} = (e_1, \ldots, e_L)$ that maximizes the probability $Pr(\mathbf{e}|\mathbf{f})$, given the source string $\mathbf{f} = (f_1, \ldots, f_M)$ (where $\mathbf{f}$ refers to French and $\mathbf{e}$ refers to English in



(a) Text-to-text translation model (b) Image-to-text translation model

Figure 4.3: Graphical models of translation models for parallel datasets. L, M are the number of items in each pair of the parallel dataset. In the text-to-text case the variables f and e denote observed French and English words. In the image-to-text case the English words have bee exchanged by a visual vocabulary that is extracted from localized features v_L that are vector quantized by I_{ev}.

the original work by Brown *et al.* (1993)). Using Bayes' rule and maximizing the numerator, the following equation is obtained:

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}}\, Pr(\mathbf{e}|\mathbf{f}) = \underset{\mathbf{e}}{\operatorname{argmax}}\, Pr(\mathbf{f}|\mathbf{e})Pr(\mathbf{e}). \tag{4.2}$$

The application of Bayes' rule incorporates $Pr(\mathbf{e})$ into the formula, which takes into account the probability that $\hat{\mathbf{e}}$ is a correct English string.

$Pr(\mathbf{f}|\mathbf{e})$ is known as the *translation model* (prediction of $\mathbf{f}$ from $\mathbf{e}$), and $Pr(\mathbf{e})$ as the *language model* (probabilities over $\mathbf{e}$ independent of $\mathbf{f}$). Most of the work that transfers this concept to image annotation tasks (Duygulu *et al.*, 2002; Wachsmuth *et al.*, 2003; Jamieson *et al.*, 2006) concentrates on the translation model; taking $\mathbf{f}$ as the words in the text and $\mathbf{e}$ as the visual words in the images, they thus predict words from image items. However, the omission of the language model component, $Pr(\mathbf{e})$ (in this case, probabilities over the "language" of images—i.e., over "good" image representations), can be seen as a shortcoming. The structural information in images is mostly neglected.

4.2.1 Parameter estimation

$Pr(\mathbf{f}|\mathbf{e})$ has a huge number of parameters, because we have to consider all possible word sequences over both languages. In order to train a translation model using parallel datasets, certain independence assumptions have to be made. A simple variant is the IBM-model-1 described by Brown *et al.* (1993),

$$Pr(\mathbf{f}|\mathbf{e}) = \sum_{\mathbf{a}} Pr(M) \prod_{j=1\ldots M} Pr(a_j|L) Pr(f_j|a_j, e_{a_j}) \tag{4.3}$$

where M is the number of French words in $\mathbf{f}$, L is the number of English words in $\mathbf{e}$, and $\mathbf{a}$ is an *alignment* that maps each French word to one of the English words, or to the "null" word e_0. $Pr(M) = \epsilon$ is constant and $Pr(a_j|L) = 1/(L+1)$ depends only on the number of English words.

Thus, the conditional probability of f_j depends only on its own alignment to an English word, and not on the translation of other words $f_{i,i\neq j}$. Alignments including mappings from and to multiple words cannot be formulated, but different French words can be mapped on the same English word. These assumptions lead to the following formulation, in which $\mathbf{t}(f_j|e_{a_j})$ defines a probabilistic translation table from English words to French words[1]

$$Pr(\mathbf{f}|\mathbf{e}) = \frac{\epsilon}{(L+1)^M} \prod_{j=1\ldots M} \sum_{a_j=0\ldots L} \mathbf{t}(f_j|e_{a_j}) \tag{4.4}$$

The graphical model in Fig. 4.3 depicts the underlying independence assumptions of the translation model. In the following, the algorithm is searching for the parameters with the maximum likelihood, i.e. the translation table that maximizes the probability of the training set $\mathcal{S} = \{(\mathbf{f}^{(s)}, \mathbf{e}^{(s)})|s = 1\ldots S\}$:

$$\hat{\mathbf{t}} = \underset{\mathbf{t}}{\operatorname{argmax}}\, score_{\mathbf{t}}(\mathcal{S}) = \underset{\mathbf{t}}{\operatorname{argmax}} \prod_{s=1}^{S} Pr(\mathbf{f}^{(s)}|\mathbf{e}^{(s)}; \mathbf{t}) \tag{4.5}$$

If the alignment $\mathbf{a} = (a_1, \ldots, a_M)$ would be given, $\mathbf{t}(f|e)$ could be simply estimated by counting the occurrences of aligned pairs (f, e). Otherwise, if the translation table would be given, we could compute the probability of an alignment (f_j, e_{a_j}). This is a classical case for applying the EM-algorithm in

[1]The transformation from Eq. 4.3 to Eq. 4.4 is given in Appendix A.1

order to get an iterative solution. Therefore, we compute the expected count $c(f|e;\mathbf{f},\mathbf{e})$ that the words f and e are aligned given by

$$c(f|e;\mathbf{f},\mathbf{e}) = \sum_{\mathbf{a}} Pr(\mathbf{a}|\mathbf{e},\mathbf{f}) \sum_{j=1...M} \delta(f,f_j)\delta(e,e_{a_j}) \tag{4.6}$$

Then, using Eq. 4.4 the EM-iteration can be formulated by[2]

$$c(f|e;\mathbf{f},\mathbf{e}) = \frac{\mathbf{t}(f|e)}{\mathbf{t}(f|e_0)+\cdots+\mathbf{t}(f|e_L)} \sum_{j=1...M} \delta(f,f_j) \sum_{i=0...L} \delta(e,e_i). \tag{4.7}$$

$$\mathbf{t}(f|e) = \lambda_e^{-1} \sum_{s=1...S} c(f|e;\mathbf{f}^{(s)},\mathbf{e}^{(s)}) \tag{4.8}$$

where S is the number of paired data-items of the training set.

In the following, we will discuss how translation models can be applied to parallel datasets of captionized images, i.e. images paired with associated text that partially refers to the image. Once a translation model has been learnt, it can be used for image annotation and region labeling. However, we cannot expect to achieve high quality annotations for arbitrary visual concepts. The feature choice critically determines which concepts can and which concepts cannot be distinguished.

4.2.2 Applying translation models to captionized images

Translation models provide an interesting perspective to the problems of object recognition and image understanding. Both processes are formulated in a coherent probabilistic framework. Translation is treated as a two-way process from visual models to words and from words to the activation of visual models. Finally, translation models can be learnt from loosely coupled parallel datasets. However, translation models work on discrete vocabularies that define meaningful chunks of data. Thus, a key question for applying translation models to visual data is that of image representation.

[2] The derivation of Eq. 4.7 is discussed in Appendix A.1.

Blob-based image representations

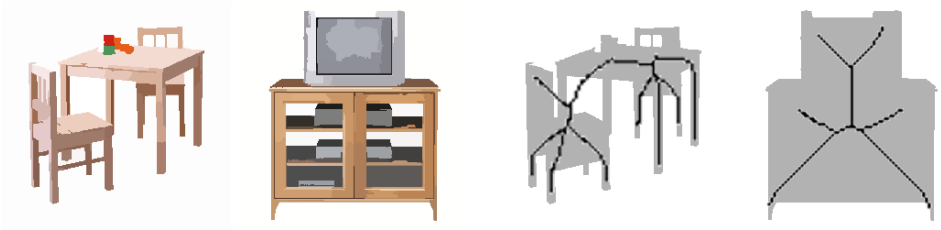
Barnard & Forsyth (2001); Barnard *et al.* (2001); Duygulu *et al.* (2002) were the first who transferred the general idea of translation models to the field of automatic image annotation. In their work, they start with an unstructured collection of images and captions which is freely available from many sources. Barnard et al. present results for the Corel-dataset that includes a large variety of scene categories from flying airplanes to tigers in the jungle (Fig. 4.2). Each image caption has about 4-5 keywords from a lexicon of 371 words. They define a visual vocabulary based on a blob-representation of the image. A region segmentation is computed using the Normalized Cuts algorithm (Shi & Malik, 2000). The largest regions (typically between 5 and 10 above a size threshold) are characterized by 33 features (region color and standard deviation, region average orientation energy (12 filters), region size, location, convexity, first moments, compactness). The feature vectors from 4500 Corel images are clustered into 500 *blobs* using k-means.

Experiments on a held-out test set of 500 images revealed large variations in the quality of annotation results (Duygulu *et al.*, 2002). 80 words from the 371 word vocabulary had a sufficiently peaked distribution in the translation table to be usable. The best result was achieved for the word 'petals' with a precision of 100% and a recall of 50%. Most other terms had a precision below 30% even for higher annotation thresholds.

Blob-based image representations necessarily face problems caused by occlusions, low contrast, and heterogeneous object surfaces leading to over- and under-segmentations. This is problematic because the translation model assumes a semantically meaningful mapping from single regions to single words. Barnard *et al.* (2003) have proposed a merging strategy based on similarly predicted annotation vectors, but this does not solve the key problem to deal with compound objects that consist of heterogeneous parts. Furthermore, the coarse shape-based features defined on regions (like compactness, convexity, or first moments) have been proved to be very weak in image annotation tasks (Barnard *et al.*, 2003). Most of the annotation performance is driven by features defined on colors and textures.

Extending translation models to compounds

Both drawbacks of blob-based translation models have been discussed by Wachsmuth *et al.* (2003). They propose two different approaches for dealing



(a) Region segmentation (b) Skeleton of merged regions

Figure 4.4: Examples of the IKEA dataset. The shock graph is computed from the skeleton of a region. Skeleton points are segmented into strokes with qualitatively similar boundary conditions, e.g. parallel boundaries. These define the nodes of the graph. The connectivity of the strokes define the edges of the graph.

with one-to-many associations between words and regions:

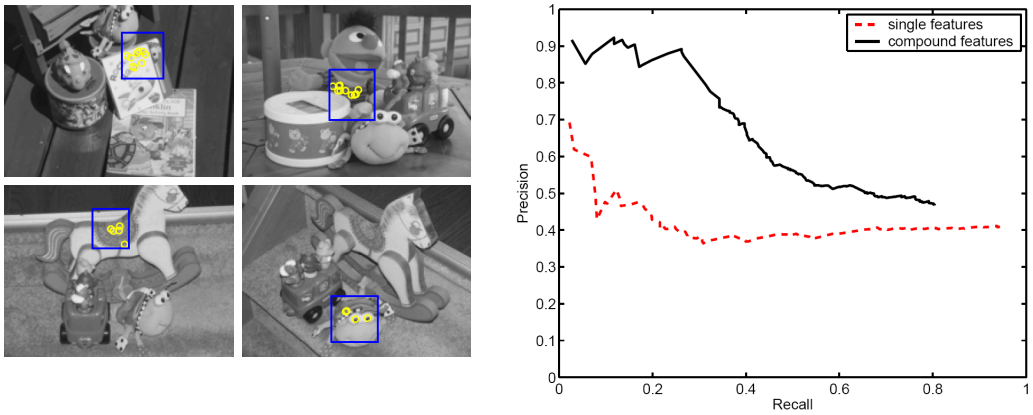
1. For accidental over-segmentations, potential merges of visual words can be judged by the change in translation score (Eq. 4.5). We iterate over the training set $\mathcal{S}$ by temporarily adding merges in $\tilde{\mathcal{S}}$,

$$score(\tilde{\mathcal{S}}^{(t)}) = \prod_{(\mathbf{f},\mathbf{e})\in\tilde{\mathcal{S}}^{(t)}} Pr(\mathbf{f}|\mathbf{e},\mathbf{t}^{(t)}) \tag{4.9}$$

   If the translation score of the data-item increases we persistently add the merged blob to the image description of the training set $\mathcal{S}^{(t+1)}$ and learn a new translation model $\mathbf{t}^{(t+1)}$.

2. For non-accidental over-segmentations, a mutual information measure proposed by Melamed (1997) in the context of text-to-text translation is used to generate structured compounds. The method of Melamed and its adaptation to image datasets will be discussed later in this section.

For including a stronger model of shape, they suggest a *shock graph* description of regions (Siddiqi *et al.*, 1999) (Fig. 4.4). These have been shown to be able to characterize the silhouettes of object categories in structural terms. However, it is quite tricky to deal with the combinatorics of possible region merges. Shapes drastically change if regions are merged and components of over-segmented regions are not descriptive enough to drive the grouping process.



(a) Example detections of learned compound features with labels 'rocket', 'ernie', 'horse', and 'bug'

(b) recall-precision graph for single and compound features

Figure 4.5: Results from Jamieson *et al.* (2006)

Bags of localized image descriptors

Point features consist of a local image descriptor that is computed for each interest point detected in an image. They are less affected by occlusions and do not require any image segmentation. Nevertheless, words will typically refer to collections of point features rather than single features. Thus, we need to extend the translation model approach for one-to-many associations. Jamieson *et al.* (2006) use the translation score to drive the grouping of SIFT features to compounds. The visual vocabulary is trained on a set of 300,000 local image descriptors that are randomly selected from a pool of 2,500 stock photo images. A k-means algorithm is applied for generating 5,000 cluster centers defining corresponding visual words $\mathcal{V} = \{v_1, \ldots, v_L\}$. A *compound feature* is essentially a 'bag' of such local features and is defined by a triple

$$c_m = \{\mathcal{V}_m, \eta_m, k_m\}, \text{ where } \mathcal{V}_m = \{v_{mj} | v_{mj} \in \mathcal{V}, j = 1 \ldots J_m\} \tag{4.10}$$

where η_m is a detection threshold, k_m is the size of a neighborhood, and J_m is the number of single features that define the compound.

A compound is detected in an image if at least η_m features out of $\mathcal{V}_m$ are found in a neighborhood of k_m localized features.

Compound features are generated in an iterative approach that follows a greedy search in the combinatorial space of possible compounds.

1. Initialize the set of compound features $\mathcal{C} = \{c_m; m = 1 \dots M\}$ with $c_m = (\{v\}, 1, k_m)$ by choosing those $n_{seed} = 20$ singletons $v \in \mathcal{V}$ for each word w with the highest likelihood $Pr(w|v)$.

2. For each compound $c_m \in \mathcal{C}$, generate one of the following modifications $\tilde{c}$: (i) add a singleton feature from the local neighborhood, (ii) eliminate a singleton feature from $\mathcal{V}_m$, (iii) change the detection threshold η_m.

3. Test the modification $\tilde{c}$ by substituting c with $\tilde{c}$ and computing a new translation model. If $Pr(w|c) < Pr(w|\tilde{c})$ then accept the modification.

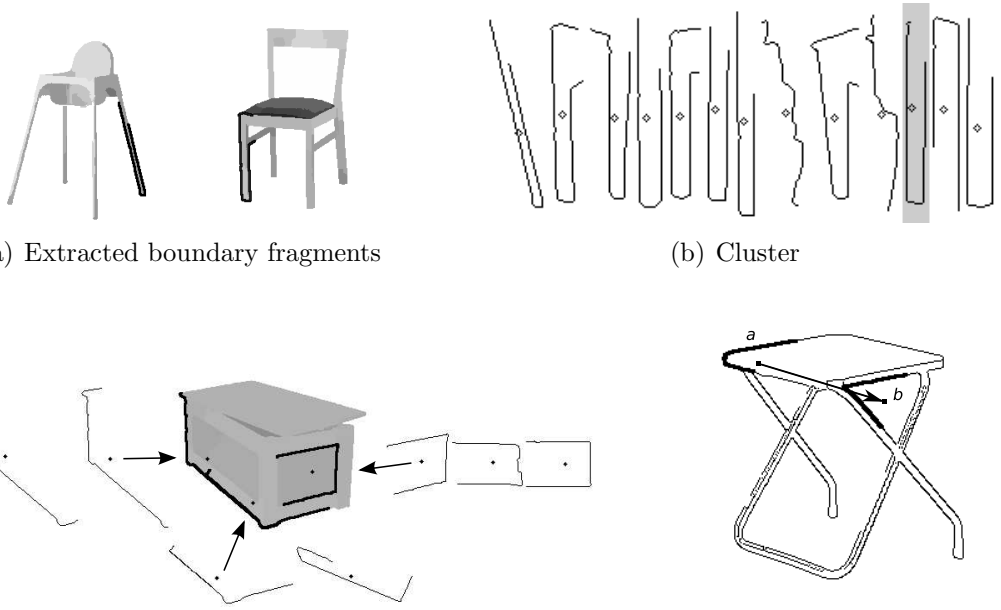
In Fig. 4.5 several compounds are shown that have been learnt on a dataset of 228 images randomly divided into 128 training images and 100 for testing. Each image includes 3 or 4 toy objects out of a pool of 10 in front of one of 15 different backgrounds. The training set had captions with the keywords of the toy objects presented in the image and an additional number of 2 to 5 random words from a pool of distractor labels. The precision-recall graph in Fig. 4.5 shows that the compound learning strategy significantly improves the annotation/retrieval results.

Structured compounds of boundary fragments

Local descriptors exploit the textural characteristics of object surfaces, but they do not have an idea of the overall shape of an object. Moringen *et al.* (2008) focus on an alternative image representation that builds on boundary fragments. These can be directly extracted from an image by a connected component analysis on edge pixels (Opelt *et al.*, 2004) or be generated from an image abstraction provided by a region segmentation. In the second case, region boundaries define the edge pixels. A boundary fragment $\mathbf{f}$ can simply be defined by a connected sequence of edge pixels $f_k = (x, y)$,

$$\mathbf{f} = (f_1, \dots, f_K), \text{ where } |p_k - p_{k+1}| \leq \sqrt{2}, \quad 1 \leq k < K, |\mathbf{f}| = K \qquad (4.11)$$

Similar to Opelt *et al.* (2004), Moringen et al. extract fragments by chaining from randomly chosen seed points. These provide templates for fault-tolerant shape recognition by using chamfer matching as described by Borgefors (1988). Chamfer matching utilizes a distance transform in order to implement an efficient way of computing the edge distance d_{edge} between a



(a) Extracted boundary fragments (b) Cluster

(c) Compound (d) Spatial relation

Figure 4.6: Building compounds from boundary fragments. First, fragments are clustered using a symmetrically defined edge distance. Secondly, compounds are learnt that encode spatial relations between fragment classes.

boundary fragment $\mathbf{f}$ possibly transformed by T and an edge image $\mathbf{I}$,

$$d_{edge}(\mathbf{f}, T, \mathbf{I}) \equiv \frac{1}{|\mathbf{f}|} \sum_{k=1}^{|\mathbf{f}|} \mathbf{I}^d[(Tf)_k]^2, \tag{4.12}$$

where $\mathbf{I}^d$ is the distance transformed image (pixels are coding the distance to the next edge pixel rather than the edge pixels themselves).

In the following, the edge distance serves two different purposes in the translation framework: (i) it provides the basis for a distance metric on boundary fragments that is used for clustering purposes and (ii) it defines a detector for fragment classes on images.

Similar to the approaches described above, a basic visual vocabulary is learnt by clustering a set of singelton features. This time an agglomerative clustering is used because the fragments and edge distance do not form a vector space[3]. Moringen *et al.* (2008) use a symmetric variant of d_{edge} for

[3]As a consequence, the mean-fragment cannot be computed, directly.

clustering,

$$\begin{aligned} d_{symm}(\mathbf{f}_1, \mathbf{f}_2) &= d'_{edge}(\mathbf{f}_1, \mathbf{f}_2) + d'_{edge}(\mathbf{f}_1, \mathbf{f}_2), \\ \text{where } d'_{edge}(\mathbf{f}_1, \mathbf{f}_2) &= \min_{T \in \mathcal{T}} d_{edge}(\mathbf{f}_1, T, \mathbf{I}_{\mathbf{f}_2}) \end{aligned} \tag{4.13}$$

Here, $\mathcal{T}$ is a discrete set of transformations that is applied to $\mathbf{f}_1$ when overlaying it over an image during chamfer matching, $\mathbf{I}_{\mathbf{f}_2}$ is a bitmap representation of the boundary fragment $\mathbf{f}_2$. Then, a cluster v_m is defined by a triple $(\mathcal{F}_l, \hat{f}_l, \eta_l)$ consisting of a set $\mathcal{F}_l$ of fragments, a representative fragment $\hat{f}_l$ and a detection threshold η_l that is later optimized on the training set by applying the translation model.

In agglomerative clustering, there are different possibilities to transfer the distance function defined on elements to a distance function between clusters. Moringen (2007) reports a good performance for a maximum operation[4],

$$d_{max}(v_1, v_2) = \max_{\mathbf{f}_j \in \mathcal{F}_1, \mathbf{f}_k \in \mathcal{F}_2} d_{symm}(\mathbf{f}_j, \mathbf{f}_k), \text{ with } v_l = (\mathcal{F}_l, \hat{f}_l, \eta_l), l = 1, 2. \tag{4.14}$$

The clusters define fragment classes $\mathcal{V}$ that provide the basic visual vocabulary for the translation model. Fig. 4.6 shows an exemplary cluster that may be associated with the semantic concept of *legs of chairs, tables, or stools*. More specific visual descriptions can be defined by visual compounds,

$$\begin{aligned} c_m = (\mathcal{V}_m, \mathcal{R}_m), \text{ where } \mathcal{V}_m &= \{v_{mj} | v_{mj} \in \mathcal{V}, j = 1 \dots J_m\}, \\ \mathcal{R}_m &= \{r_m^{jk} : \mathcal{P} \times \mathcal{P} \to \mathbb{R} | j, k = 1 \dots J_m\}. \end{aligned} \tag{4.15}$$

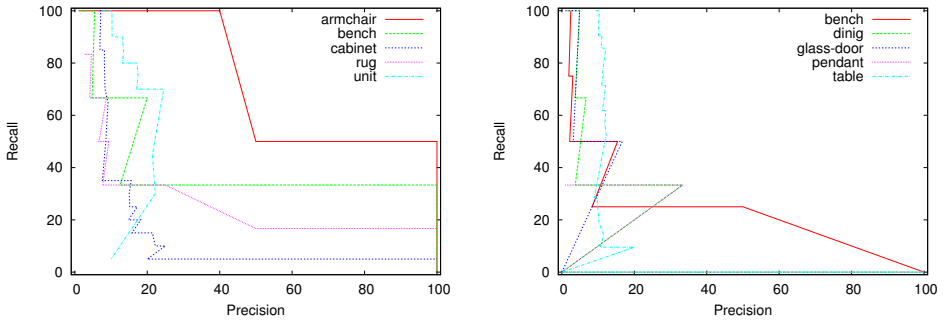
Here, $\mathcal{V}_m$ is a collection of fragment classes with spatial relations $\mathcal{R}_m$ between them. Let $v_{m\,1}$ and $v_{m\,2}$ be fragment classes detected in the image at positions $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$. Then the spatial relation between them can be judged by

$$r_m^{1\,2}(p_1, p_2) = \frac{1}{n_m} \sum_{i=1}^{n_m} \mathcal{N}_{\mu_{mi}^{1\,2}, \sigma}(p_2 - p_1) \tag{4.16}$$

Here, n_m is the number of occurrences of the compound c_m in the training set. Each offset between the detected fragment classes v_{mj} and v_{mk} is stored in μ_{mi}^{jk} defining a Gaussian kernel with standard deviation σ.

During the training of the translation model, Moringen et al. search for compounds by using Melamed's method for finding non-compositional compounds (NCCs) in parallel text (Melamed, 1997) (see 4.4.2). Fig. 4.6

[4] In cluster analysis, the max-method is also called *complete-linkage*.



(a) Training set (b) Test set

Figure 4.7: Boundary fragment compounds: results for a furniture dataset with 300 training images and 225 test images.

shows an exemplary compound generated in an experiment on a captionized furniture dataset. The dataset consisted of 525 images (300 training, 225 test) with single pieces of furniture or groups of furniture. The captions have been processed by a tagger (Brill, 1994) and partial parser (Abney, 1991, 1996) leaving between 1 and 4 head nouns. Precision-recall curves are given in Fig. 4.7 for some of the vocabulary words learnt. Relatively low precision values indicate that there is a large variance of shapes in the dataset. The training set included only a few exemplars per word category so that generalizing models are difficult to learn. However, for some words like 'bench' reasonable compound models have been extracted.

4.3 Co-occurrence statistics

Another way to deal with parallel datasets is to transform them to *co-occurrence* data (COD) and to learn co-occurrence statistics on the transformed dataset.

Definition 2 (Co-occurrence data) *In the general setting of COD, we have two vocabularies (finite sets)* $\mathcal{X} = \{x_1, \ldots, x_N\}$ *and* $\mathcal{Y} = \{y_1, \ldots, y_M\}$. *Elementary observations consist of pairs* $(x_i, y_j) \in \mathcal{X} \times \mathcal{Y}$, *i.e. a joint occurrence of words or abstract objects of both vocabularies. A sample set of COD is given by a collection of elementary observations with an arbitrary ordering*

$$\mathcal{S} = \{(x_{i(s)}, y_{j(s)}, s); s = 1 \ldots S, i(s) \in [1, N], j(s) \in [1, M]\}$$

The information in $\mathcal{S}$ is completely characterized by its sufficient statistics $n_{ij} = |\{(x_i, y_j, s) \in \mathcal{S}\}|$ which is the frequency of the joint observation of (x_i, y_j).

A COD can be easily constructed from a parallel dataset. In a first approach, a parallel dataset is treated as a collection of documents $\mathcal{X}$ and a set of (multi-vocabulary) terms $\mathcal{Y} \equiv \mathcal{E} \cup \mathcal{F}$. A stochastic process generates co-occurrences $(x, y) \in \mathcal{X} \times \mathcal{Y}$ of documents and terms which can be interpreted as terms in that document. In this case, the parallel structure of data-items is ignored and is transformed into single concatenated sequences of symbols. Furthermore, the ordering information or structure between symbols in each document is ignored.

In a second approach, we abstract from the sequential nature of the data and just model if a specific term occurs in the data or not. Given the vocabularies $\mathcal{E}$ and $\mathcal{F}$ of the parallel dataset, we just define $\mathcal{X} \equiv \mathcal{E}$, $\mathcal{Y} \equiv \mathcal{F}$, and introduce indicator variables X_i and Y_i for each term $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$, respectively. Then, for each parallel data-item $(\mathbf{e}^{(s)}, \mathbf{f}^{(s)})$ a set of pairs of indicator variables can be specified characterizing co-occurring term-pairs,

$$(\mathbf{e}^{(s)}, \mathbf{f}^{(s)}) \rightarrow \{(X_i^{(s)}, Y_j^{(s)})\}_{i \in [1 \ldots |\mathcal{X}|], j \in [1 \ldots |\mathcal{Y}|]} \tag{4.17}$$

$$\text{where } X_i^{(s)} = \begin{cases} 1, & \text{if } x_i \in \mathbf{e}^{(s)} \\ 0, & \text{otherwise} \end{cases}, \quad Y_j^{(s)} = \begin{cases} 1, & \text{if } y_j \in \mathbf{f}^{(s)} \\ 0, & \text{otherwise.} \end{cases} \tag{4.18}$$

The sample set of COD is, then, given by those pairs $\{(x_{i(s)}, y_{j(s)}, s); s = 1 \ldots S'\}$ where both corresponding indicator variables have the value 1. Thus, the model abstracts from the correct alignment and considers valid as well as invalid pairings of both vocabularies. Invalid meaningless pairings add noise to the co-occurrence statistics.

The main challenge of modeling COD is the problem of *data sparseness*, i.e. for large vocabularies the majority of pairs (x_i, y_j) are rarely observed or even not observed at all. There are a bunch of methods for dealing with this kind of deficiency ranging from smoothing techniques (Katz, 1987; Chen & Goodman, 1996), model interpolation with held out data (Jelinek, 1985; Jelinek & Mercer, 1980), to cluster methods (Griffiths *et al.*, 1986; Rijsbergen, 1979) and improved feature representations (Salton, 1971; Deerwester *et al.*, 1991).

4.3.1 Mixture models and clustering methods

A frequent approach to modeling joint distributions $Pr(X = x_i, Y = y_j)$ is to assume that both symbols are generated from a common source $C \in \mathcal{C} = \{c_1, \ldots, c_K\}$. This can be modeled by a hidden variable C resulting in a symmetric mixture model (SMM) shown in Fig. 4.8(a). The symmetric SMM can also be transformed into an asymmetric parameterization (Fig. 4.8(b)). In this case, X is interpreted as a mixture of components C and the components are interpreted as a partitioning of the $\mathcal{Y}$-space.

$$Pr(\mathcal{S}|p, r, q) = \prod_{(x_i^{(s)}, y_j^{(s)}) \in \mathcal{S}} p_i \sum_k q_{j|k} r_{k|i} \tag{4.19}$$

Mixture models are tightly related to cluster models which introduce a deterministic assignment $I : \mathcal{X} \to \mathcal{C}$ of elements $x \in \mathcal{X}$ to components or clusters $c \in \mathcal{C}$ (Fig. 4.8(c)). In the following, we use the index notation I_{ki} for $I(x_i) = c_k$. Because only the $\mathcal{X}$-space is deterministically partitioned into clusters while the $\mathcal{Y}$ space is modeled by a mixture of probability distributions, this model is named *Asymmetric Cluster Model* (ACM). Parameters are estimated by an EM-style algorithm with

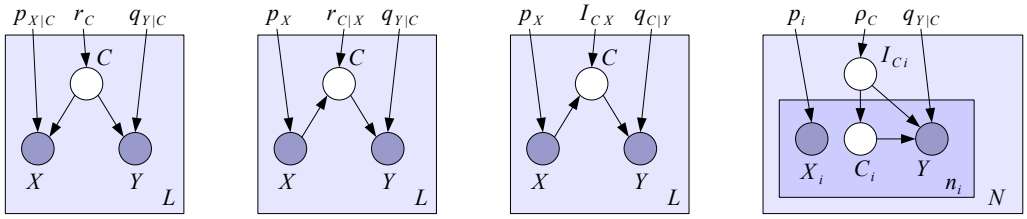
$$\hat{p}_i = n_i / L \tag{4.20}$$

$$\hat{I}_{ki} = \begin{cases} 1 & \text{if } k = \arg\min_\nu D[n_{j|i} | \hat{q}_{j|\nu}] \\ 0 & \text{otherwise,} \end{cases} \tag{4.21}$$

$$\hat{q}_{j|k} = \frac{\sum_{i=1}^N \hat{I}_{ki} n_{ij}}{\sum_{i=1}^N \hat{I}_{ki} n_i} = \sum_{i=1}^N \frac{\hat{I}_{ki} n_i}{\sum_{h=1}^N \hat{I}_{kh} n_h} n_{j|i}. \tag{4.22}$$

Here, n_i and n_{ij} are counts of occurrences of x_i and y_j in the training set $\mathcal{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_L\}$, $n_{i|j}$ is the histogram of (x_i, y_j) co-occurrences normalized with regard to y_j. $D[n_{j|i} | \hat{q}_{j|\nu}]$ is a distortion measure (cross entropy or Kullback-Leibler divergence) between the model parameters and the empirical distribution.

Hofmann & Puzicha (1998) present a probabilistic ACM for co-occurrence data. The graphical model is shown in Fig. 4.8(c). They take a Bayesian perspective on the ACM and represent the clustering parameters as random variables $I_{ki}, i = 1 \ldots N$ that bind all occurrences of $X^{(s)} = x_i, s = 1 \ldots L$. The clustering of the $\mathcal{X}$-space assumes that all n_i occurrences are generated by a common class $C_i = c_k$ that is determined by I_{ki}. Then, $y_j^{(s)}$ is generated



(a) sym. SMM (b) asym. SMM (c) ACM (d) prob. ACM

Figure 4.8: Mixture models: The symmetric and asymmetric Separated Mixture Models (SMMs) represent the same model assumptions. The asymmetric cluster model (ACM) exchanges the probabilistic assignment of a mixture components $r_{C|X}$ by a deterministic cluster assignment I_{CX}. In the probabilistic ACM the parameters I_{CX} is treated as a random variable. For the graphical model the L data-items are organized in N groups ($N = |\mathcal{X}|$) with n_i members. In each group i all variables X have an observed value of x_i. Thus, the variable C_i in the inner plate does not depend on X_i any more.

from a distribution with parameters $q_{j|k}$. The joint probability of a training set $\mathcal{S}$ and cluster variables I is given by

$$Pr(\mathcal{S}, I|\rho, p, q) = Pr(\mathcal{S}|I, p, q)\ Pr(I|\rho), \quad \text{where } Pr(I|\rho) = \prod_{i=1}^{N} \rho_k^{I_{ki}}. \quad (4.23)$$

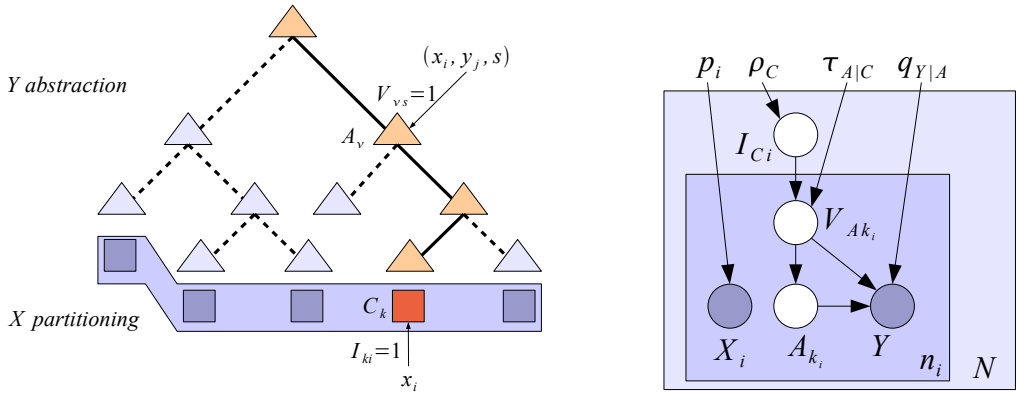
In the Bayesian framework, the probability of a new item $\mathbf{s} = (x_i^{(s)}, y_j^{(s)}, L{+}1)$ is computed from an already observed dataset $\mathcal{S}$,

$$\begin{aligned} Pr(x_i^{(s)}, y_j^{(s)}|\mathcal{S}, \rho, p, q) &= \sum_{k=1}^{K} Pr(\mathbf{s}|I_{ki} = 1, \mathcal{S}_i, p, q) Pr(I_{ki} = 1|\mathcal{S}_i, \rho, q) \\ &= p_i \sum_{k=1}^{K} q_{j|k} \langle I_{ki} \rangle. \end{aligned} \quad (4.24)$$

where $\langle I_{ki} \rangle$ denotes the posterior of the indicator variables.

In the EM-scheme Eq. 4.21 is replaced by computing the posterior probabilities

$$\langle I_{ki} \rangle^{(L+1)} = \frac{Pr(\mathcal{S}_\rangle|I_{ki} = 1, \hat{\rho}^{(L)}, \hat{q}^{(L)})}{Pr(\mathcal{S}_\rangle|\hat{\rho}^{(L)}, \hat{q}^{(L)})} = \frac{\hat{\rho}_k^{(L)} \prod_{j=1}^{M} \left\{ \hat{q}_{j|k}^{(L)} \right\}^{n_{ij}}}{\sum_{\nu=1}^{K} \hat{\rho}_\nu^{(L)} \prod_{j=1}^{M} \left\{ \hat{q}_{j|\nu}^{(L)} \right\}^{n_{ij}}} \quad (4.25)$$



(a) hierarchical clustering (b) graphical model

Figure 4.9: Hierarchical asymmetric clustering model (HACM). First one of the clustering nodes k_i is selected which partition the $\mathcal{X}$-space. This determines a path $A_{k_i}^{(s)}$ in the tree of abstraction nodes from which one abstraction node $A_\nu^{(s)}$ is selected. This node deterministically generates the $x_i^{(s)}$ symbol and probabilistically generates the $y_j^{(s)}$ symbol.

The priors $\hat{\rho}_k^{(L)}$ are computed by

$$\hat{\rho}_k^{(L)} = \frac{1}{N} \sum_{i=1}^{N} \langle I_{ki} \rangle^{(L)} \tag{4.26}$$

Based on the ACM, Hofmann and Puzicha define a hierarchical clustering model, called HACM (Hofmann & Puzicha, 1998). The general scheme and graphical model is depicted in Fig. 4.9. The x_i symbols are still generated from a common class $C_i = c_k$ which is determined by the clustering variables $I_{ki} \in \{0,1\}, i = 1 \ldots N, k = 1 \ldots K$. However, instead of generating y_j directly from these clusters, different abstraction levels are introduced by successively joining clusters over the $\mathcal{X}$ space. This defines a tree structure with the clusters as leave nodes which is encoded in the coefficients $\tau_{A|C}$. In the generative model, a path in the tree is selected by the cluster c_k. The variable A_k selects a node on the abstraction path that generates a data-item (x_i, y_j, s). Due to the abstraction hierarchy in the tree the variables I and V are dependent

$$\sum_k \sum_{A_\nu \uparrow C_k} I_{i(s)k} V_{s\nu} = 1, \forall s \in \{1, \ldots, S\}. \tag{4.27}$$

where $A_\nu \uparrow C_k$ denotes all nodes A_ν in the tree 'above' C_k.

The node that generates (x_i, y_j, s) must lie on the path to cluster c_k determined by the cluster variable I_{ik}. The parameters of the models can be computed by a similar EM-style algorithm as discussed for the ACM. The predictive probabilities for a new data-item $\mathbf{s} = (x_i, y_j, L+1)$ is again computed from the history of data-items seen before,

$$Pr(\mathbf{s}|\mathcal{S}, \theta) = p_i \sum_\nu p_{\nu|i} q_{j|\nu} \quad \text{with } p_{\nu|i} \equiv \sum_k \langle I_{ki} \rangle \tau_{\nu|k} \tag{4.28}$$

where $\theta = (p, q, \rho, \tau)$ are the parameters of the model. The prior $\tau_{\nu|k}$ can also be made dependent of the specific x_i symbol additional to the cluster ($\tau_{nu|k,i}$), which is typically not done.

The hierarchical asymmetric cluster model picks up the intuition that most documents – like images or text – are mixtures of more abstract and more specific terms. For example, a scene of a tropical beach sunset includes general elements like a red-colored sky and a sea up to the horizon, which are shared with many other image classes, but also palm trees which are more specific elements.

4.3.2 Likelihood ratio testing

Likelihood ratio testing builds on the second approach of transforming parallel datasets into co-occurrence statistics, i.e. each data item $(\mathbf{x}^{(s)}, \mathbf{y}^{(s)})$ is represented as a set of pairs of indicator variables

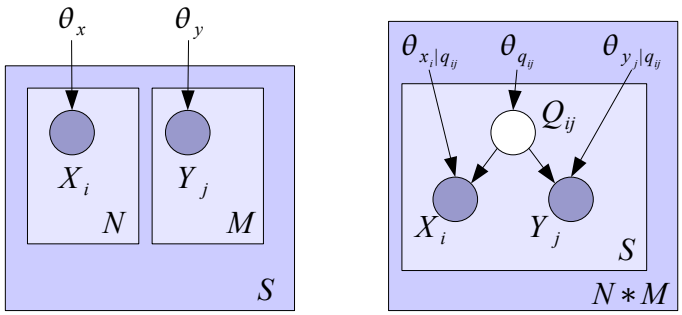
$$(\mathbf{x}^{(s)}, \mathbf{y}^{(s)}) \rightarrow \{(X_i^{(s)}, Y_j^{(s)})\}_{(i,j) \in [1 \ldots N] \times [1 \ldots M]} \tag{4.29}$$

where $\mathcal{X} = \{x_1, \ldots, x_N\}$ and $\mathcal{Y} = \{y_1, \ldots, y_M\}$ are the two vocabularies. The indicator variables $X_i^{(s)}, Y_j^{(s)} \in \{0, 1\}$ indicate the presence of the symbols x_i and y_j in the parallel data item $(\mathbf{x}^{(s)}, \mathbf{y}^{(s)})$.

We are interested in pairings (x_i, y_j) which frequently occur in different items of the dataset. The transformation ignores multiple instances of a vocabulary word and treats each (x_i, y_j) pair independently.

Likelihood ratio testing measures a model assumption against a null-hypothesis. In our case, the null-hypothesis $\mathcal{H}_0$ is an independence assumption between the two indicator variables of an (x_i, y_j)-pair,

$$Pr(X_i, Y_j|\theta_0, \mathcal{H}_0) \equiv Pr(X_i|\theta_x)\ Pr(Y_j|\theta_y), \quad \text{where } \theta_0 = (\theta_x, \theta_y). \tag{4.30}$$



(a) Null-hypothesis (b) Test-hypothesis

Figure 4.10: Graphical models for generating the co-occurrence data. The null-hypothesis makes an independence assumption, while the test hypothesis models a common source of both symbols.

The test hypothesis $\mathcal{H}_1$ consists of a joint distribution of $Pr(X_i, Y_j|\theta_1, \mathcal{H}_1)$ that includes a dependence between the two indicator variables. Specifically, it is assumed that both symbols are generated from a common source q_{ij}. An indicator variable $Q_{ij}^{(s)}$ models the presence of the common source for each item of the dataset,

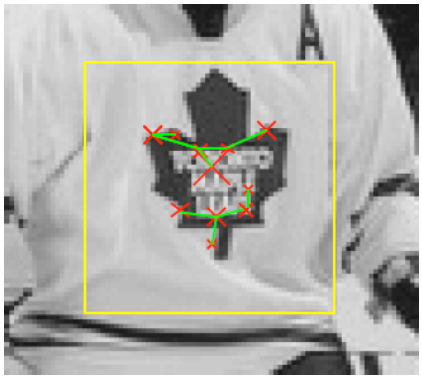
$$Pr(X_i, Y_j|\theta_1, \mathcal{H}_1) \equiv Pr(X_i|Q_{ij}, \theta_{xq})\ Pr(Y_j|Q_{ij}, \theta_{yq})\ Pr(Q_{ij}|\theta_q) \tag{4.31}$$

where $\theta_1 = (\theta_{xq}, \theta_{yq}, \theta_q)$.

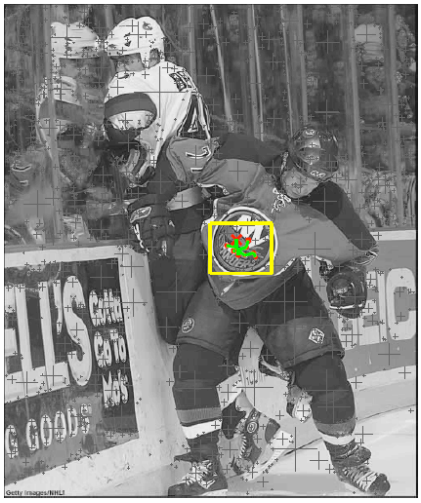
The competing model assumptions of the test- and null-hypotheses are depicted as graphical models in Fig. 4.10. Note that the test hypothesis is independently judged for each combination of x_i and y_j. Both joint distributions of the two hypotheses have free parameters that need to be estimated on the dataset using the maximum-likelihood principle. Then, the log-likelihood ratio is computed for each pair (x_i, y_j) as a correlation measure on the dataset,

$$Corr(x_i, y_j) = \log \frac{\prod_s\ Pr(X_i^{(s)}, Y_j^{(s)}|\mathcal{H}_1)}{Pr(X_i^{(s)}, Y_j^{(s)}|\mathcal{H}_0)} \tag{4.32}$$

$$= \log \frac{\prod_s\ \sum_{q\in\{0,1\}}\ Pr(Q_{ij}^{(s)} = q)\ Pr(X_i^{(s)}|Q_{ij}^{(s)} = q)\ Pr(Y_j^{(s)}|Q_{ij}^{(s)} = q)}{\prod_s\ Pr(X_i^{(s)})\ Pr(Y_j^{(s)})} \tag{4.33}$$



(a) Compound model associated with 'Toronto Maple Leafs'. Observed local descriptors are marked in red, spatial relations in green.



(b) New York **Islanders**' defenseman Alexei Zhitnik mashes Vancouver Canucks' right wing Todd Bertuzzi into the glass.

Figure 4.11: Structured compound models are learnt from a parallel dataset of captionized images. The grouping process is driven by a correlation measure between image entities and caption words. Examples taken from Jamieson *et al.* (2007).

One also can include pre-knowledge by fixing some of the parameters in θ_1. Jamieson *et al.* (2007) apply this association model to parallel data of captionized images. The caption words are modeled by variables X_i and the visual entities detected in the image are represented by Y_j. The parameters $\theta_{x_i q_{ij}} = Pr(X_i|Q_{ij})$ and $\theta_{y_j q_{ij}} = Pr(Y_j|Q_{ij} = 0)$ are given fixed values in order to set an appropriate prior for a strong association model,

$$\begin{aligned} &Pr(X_i = 1|Q_{ij} = 1) = 0.95; \\ &Pr(X_i = 1|Q_{ij} = 0) = 0.05; \qquad Pr(Y_j = 1|Q_{ij} = 0) = 0.01 \end{aligned} \tag{4.34}$$

The first two parameters strengthen the association between the caption word x_i and the corresponding common sources q_{ij}. The third parameter states that if a visual entity is detected there also should be a corresponding word in the caption.

Jamieson *et al.* (2007) use this method to drive the grouping process of a structured appearance model. It builds on the work presented in Jamieson

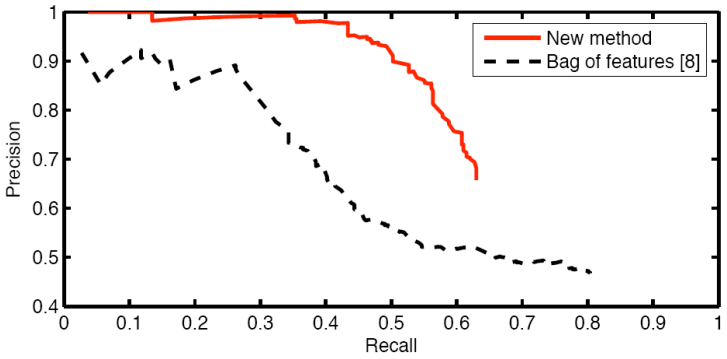


Figure 4.12: Annotation results of Jamieson *et al.* (2007) on the same data set as presented in Fig. 4.5. The bag-of-features results are generated by the method of Jamieson *et al.* (2006).

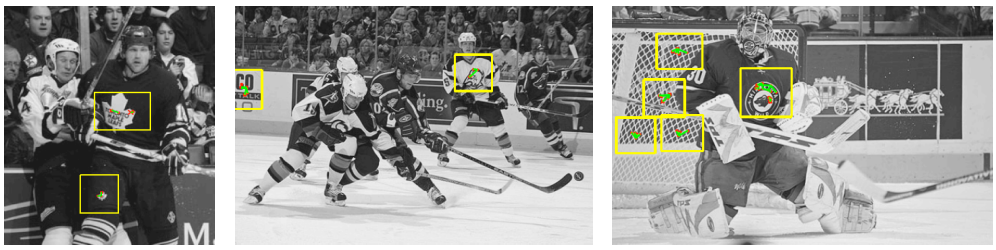
et al. (2006) which uses local descriptors of interest points as an image representation (4.2.2). Here, they use PCA-SIFT features (Ke & Sukthankar, 2004) and vector-quantize them into a set of visual words. However, the former model had a couple of shortcomings that are improved in the newer approach:

1. Instead of starting with singleton features, they start from pairs of visual features. This provides a stronger initial association between words and visual entities.

2. Instead of using a bag-of-features approach, they use structured appearance models that include spatial relations between local image descriptors. This significantly reduces the false positive rate of visual entities detected in the image and, thereby, strengthens their associations to caption words.

3. Instead of computing a full translation model, they apply a cheaper log-likelihood ratio test $Corr(x_i, y_j) > \eta$ for judging newly generated compounds. At the initial stage compounds are judged with a threshold of $\eta = 0$, which is improved during the iterative compound search. In the annotation mode, only models above a threshold of $\eta = 10$ are used for generating caption words.

The search for good seed-models is a critical step, because the following strategy for finding structured compounds applies a greedy heuristic by iteratively improving the log-likelihood ratio. Therefore, Jamieson et al. apply a



(a) Variations in scale (b) Minnesota Wild Arena (c) Detection of 'vs'

Figure 4.13: Annotation results of Jamieson *et al.* (2007). The vision compounds referring to ice-hockey team labels are automatically discovered by the learning algorithm. (b) shows an annotation result of the system. The word 'arena' is generated from the compound detected on the board left in the image. Compounds that are associated with 'vs' are shown in (c). The precision recall curve shows the increased precision by introducing spatial relationships between image descriptors.

method for finding re-occurring neighborhood patterns as proposed by Sivic & Zisserman (2004). In short, this method defines a neighborhood based on the N nearest interest points detected and characterizes each neighborhood by a sparse vector of indicator variables for each visual word. These vector representations are clustered forming re-occurring neighborhood patterns p_k. Jamieson et al. score detected neighborhood patterns by the log-likelihood ratio $Corr(x_i, p_k)$ and select frequent pairs in these neighborhoods with a consistent spatial relation as seed models.

The performance increase by introducing spatial relationships between the image descriptors is shown in Fig. 4.12. Here the same dataset was used as in Jamieson *et al.* (2006) consisting of 200 training images with captions and 125 test images. Further results have been computed on a more difficult dataset of images of National Hockey League (NHL) players and games. These were downloaded from various web sites and include associated captions like shown in Fig. 4.11(b). In order to increase potential associations with team names and team logos, city names and team names are treated as the same token. Other constraints have not been made. The image set was randomly divided into 850 training images with captions and 390 test images for annotation. About a third of the captions are full sentence descriptions whereas the remainder simply named the two teams involved in the game. As images in Fig. 4.13 show, team logos are automatically discovered by the

approach and associated with correct team names.

4.4 Mutual information methods

The log-likelihood ratio already compared two different hypotheses with regard to how well these could explain the observed data. The null hypothesis assumed an independence between the occurrences of the symbols x_i and y_j while the test hypothesis assumed them to be statistically dependent. A similar approach is taken by mutual information methods. Given a parallel dataset with two vocabularies $\mathcal{X} = \{x_1, \ldots, x_N\}$ and $\mathcal{Y} = \{y_1, \ldots, y_M\}$, we need to automatically identify strong associations between symbols x_i and y_j. If x_i is observed in the first part of a data-item, there should be a strong expectation to find the associated y_j in the second part of the data-item. This kind of predictability is measured by the mutual information between two random variables, in our case between the two indicator variables X_i and Y_j,

$$I(X_i;Y_j) = H(X_i) - H(X_i|Y_j) = H(Y_j) - H(Y_j|X_i) \tag{4.35}$$

$$= H(X_i) + H(Y_j) - H(X_i, Y_j). \tag{4.36}$$

Eq. 4.35 states that knowledge on X_i reduces the entropy of the *a posteriori* distribution of Y_j (given by $H(Y_j|X_i)$) compared to the *a priori* distribution of Y_j (given by $H(Y_j)$) – and vice versa – by a certain amount of $I(X_i;Y_j)$ called *mutual information*. Thus, strong associations of x_i and y_j correspond to a high values of $I(X_i;Y_j)$, independent occurrences of x_i and y_j correspond to a mutual information of $I(X_i;Y_j) = 0$. As shown in the Appendix A.2, the mutual information can be calculated by

$$I(X_i;Y_j) = \sum_{x\in\{0,1\}} \sum_{y\in\{0,1\}} Pr(X_i = x, Y_j = y) \log \frac{Pr(X_i = x, Y_j = y)}{Pr(X_i = x)Pr(Y_j = y)} \tag{4.37}$$

This provides a basic measurement that discriminates between random and systematic co-occurrences of vocabulary words x_i and y_j in parallel datasets.

4.4.1 Learning an audio-visual lexicon

Roy & Pentland (2002) apply a mutual information measure for learning an audio-visual lexicon from noisy acoustic input and color images. Both the

vocabulary of words and the set of visual concepts are not known before and are acquired in parallel. The acoustic input consists of natural multi-word utterances. Therefore, the word learning task includes the segmentation of the acoustic input into words. The figure background segmentation on the vision side is simplified by using a uniform background and by avoiding occlusions.

In the training phase the correspondence problem is solved by pointing on the referenced object or presenting a single object to the system. A phoneme recognizer that consists of an all-phoneme loop hidden Markov model (HMM) and a phoneme transition bigram calculates the most likely phoneme trace from the acoustic input. It achieves a phoneme recognition accuracy of about 70%. The visually observed objects are separated from the background and are characterized by a color and a shape histogram. The combined phoneme trace and histograms of the presented object are subsequently called acoustic-visual events (AV-events). First, a sufficient number of AV-events is accumulated. For each AV-event several word hypotheses are extracted by variable splitting of the phoneme trace. The word-object pairs are reduced by several filter criteria, like prosodic highlight, recurrence of speech segments, etc.

In order to find final word-shape or word-color clusters that constitute new words and their visual categorical meanings, Roy and Pentland introduce separate distance measures between visual events and between acoustic events that are combined using a mutual information measure.

The distance between two speech segments a, b is defined on the basis of a probability measurement. The phoneme recognizer calculates the most likely phoneme sequences $Q_{a/b}$ of the segments a respectively b. From these, specific HMMs $\lambda_{a/b}$ are generated using the phonemes as states and connecting them in a strictly left to right manner. State transition probabilities are inherited from context-independent phoneme models. The distance is based on the "cross"-production probabilities, that is the probability that Q_a is produced by the HMM λ_b and vice versa:

$$d_A(a,b) = -\frac{1}{2}\left\{\log\left[\frac{P(Q_a|\lambda_b)}{P(Q_a|\lambda_a)}\right] + \left[\frac{P(Q_b|\lambda_a)}{P(Q_b|\lambda_b)}\right]\right\} \tag{4.38}$$

where $Q_{a/b}$ is the phoneme sequence of a resp. b and
$\lambda_{a/b}$ is the HMM derived from the speech segment a resp. b.

The distance of two visual events is measured by the χ^2 divergence of the

associated histograms $\mathbf{x}, \mathbf{y}$:

$$d_V(\mathbf{x}, \mathbf{y}) = \chi^2(\mathbf{x}, \mathbf{y}) = \sum_i \frac{(x_i - y_i)^2}{x_i + y_i} \tag{4.39}$$

The calculation of the mutual information measure of a word-shape pair depends on two variable thresholds defining a cluster around the audio-visual event. Two variables $A, V \in \{0, 1\}$ indicate the resulting membership of other word-shape hypotheses of the cluster. The two thresholds are optimized using the maximum mutual information (MMI) $I(A; V)$ as a criterion:

$$I(A; V) = \sum_{s \in \{0,1\}} \sum_{t \in \{0,1\}} P(A = s, V = t) \log \left[\frac{P(A = s, V = t)}{P(A = s) P(V = t)} \right] \tag{4.40}$$

The selection of the final word-shape clusters is performed using a greedy strategy. Successively, the hypotheses with the highest MMI is selected and all other hypotheses which match an optimized cluster both visually and acoustically are deleted. In a final step remaining clusters are selected according to a threshold applied to the mutual information score of the cluster.

In experiments, this learning strategy turned out to be very robust and effective. Its most powerful characteristic is the generic representation of visual objects and words. No previous modeling and no manual adaptation to new domains is needed. Roy (2003) even present a first step towards syntax learning in that they generate a co-occurrence statistics of the acoustic entries in the audio-visual lexicon that is used in speech recognition. However, the aim of a bootstrapping speech and image understanding system is quite far away.

4.4.2 Learning non-compositional compounds

The predictive power of a translation model or a COD model depends on what the model is meant to predict. The translation model can only be as good as the vocabulary choice on both sides of the parallel datasets permits. In linguistics as well as in computer vision, this is the problem of finding an appropriate grouping of words or features. Non-compositional compounds (NCCs) are frequently found in language. These are compound words whose meaning cannot be synthesized from the meanings of their component words, e.g. “kick the bucket” or “hot dog”. Image representations using a visual

Parallel dataset

English half	French half	s
balance	équilibre	1
sheet	feuille	2
balance sheet	bilan	3

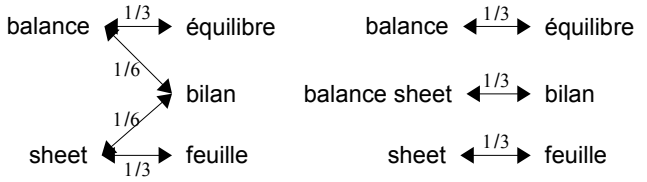


Figure 4.14: A trivial parallel text and two optional translation models modeling the joint distribution of words $Pr(X, Y)$. The first model does not know about compounds, the second one includes the NCC 'balance sheet'.

word approach have similar difficulties because, typically, there is no one-to-one correspondence to semantically meaningful concepts. Instead association models need to be based on many-to-one correspondence models like in (Jamieson *et al.*, 2006, 2007; Moringen *et al.*, 2008).

Melamed (1997) formulated a method for finding NCCs based on a mutual information measure. The objective function is defined on the basis of an already learnt translation model $Pr(x, y) = Pr(x|y)Pr(y)$ where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ are words of the two vocabularies $\mathcal{X}$ and $\mathcal{Y}$ of the languages, respectively. The mutual information indicates how well the model can predict the distribution of words in the target text given the distribution of words in the source text, and vice versa,

$$I(X;Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} Pr(X = x, Y = y) \log \frac{Pr(X = x, Y = y)}{Pr(X = x)Pr(Y = y)} \tag{4.41}$$

Melamed gives a simple example that demonstrates that the mutual information of a translation model increases if NCCs are introduced (Fig. 4.14). In the first incorrect model the translation probability mass of the word 'bilan' is distributed between the translations 'sheet' and 'balance'

$$Pr(X = \text{'balance'}|Y = \text{'bilan'}) = Pr(X = \text{'sheet'}|Y = \text{'bilan'}) = 0.5, \tag{4.42}$$

which is the same as the *a priori* distribution of X. Thus, the word 'bilan' does not contribute any information on X. In the second correct model, all probability mass is concentrated on the compound

$$Pr(X = \text{'balance sheet'}|Y = \text{'bilan'}) = 1.0 \tag{4.43}$$

Thus, we have complete information on X given the word 'bilan'. The computation of the mutual information in both cases is given in Fig. 4.15. In

In the first case the mutual information is calculated as

$$\begin{aligned} I(X;Y) =& Pr(\text{`balance'}, \text{`équilibre'}) \log \frac{Pr(\text{`balance'}, \text{`équilibre'})}{Pr(\text{`balance'})Pr(\text{`équilibre'})} \\ &+ Pr(\text{`balance'}, \text{`bilan'}) \log \frac{Pr(\text{`balance'}, \text{`bilan'})}{Pr(\text{`balance'})Pr(\text{`bilan'})} \\ &+ Pr(\text{`sheet'}, \text{`bilan'}) \log \frac{Pr(\text{`sheet'}, \text{`bilan'})}{Pr(\text{`sheet'})Pr(\text{`bilan'})} \\ &+ Pr(\text{`sheet'}, \text{`feuille'}) \log \frac{Pr(\text{`sheet'}, \text{`feuille'})}{Pr(\text{`sheet'})Pr(\text{`feuille'})} \\ =& \frac{1}{3} \log \frac{1/3}{1/2\ 1/3} + \frac{1}{6} \log \frac{1/6}{1/2\ 1/3} + \frac{1}{6} \log \frac{1/6}{1/2\ 1/3} + \frac{1}{3} \log \frac{1/3}{1/2\ 1/3} \\ =& \frac{1}{3}(\log 2 + \log 2) = \frac{2}{3} \end{aligned} \tag{4.44}$$

In the second case, we get

$$I(X;Y) = \frac{1}{3} \log \frac{1/3}{1/2\ 1/3} + \frac{1}{3} \log \frac{1/3}{1/2\ 1/3} + \frac{1}{3} \log \frac{1/3}{1/2\ 1/3} = 1 \tag{4.45}$$

Figure 4.15: Computation of the mutual information of the example presented in Fig. 4.14. Note that the word 'bilan' does not contribute any information in the first case.

principle, one needs to compute a trial translation model for each combination of NCCs and compare the resulting mutual information with that of the current base translation model. Given the huge number of potential compounds in large text corpora this is an unfeasible computational task. Melamed suggest a number of steps and assumptions that simplify this task and make it practically treatable.

As a first step, we can re-organize Eq. 4.41 as a sum of contributions of each word x of the source vocabulary $\mathcal{X}$,

$$I(X;Y) = \sum_{x \in \mathcal{X}} i^{\mathcal{Y}}(s), \text{ where } i^{\mathcal{Y}}(s) = \sum_{y \in \mathcal{Y}} Pr(x,y) \log \frac{Pr(x,y)}{Pr(x)Pr(y)}. \tag{4.46}$$

The *predictive value function* $i^{\mathcal{Y}}(x)$ enables a local treatment of vocabulary changes in $\mathcal{X}$ under the following assumption.

Assumption 1: *Treating a bigram $\langle x_i, x_j \rangle$ as an NCC will not affect the predictive value function of any $x \in \mathcal{X}$ other than x_i, x_j, and the NCC $x_{ij} \equiv \langle x_i x_j \rangle$.*

Under Assumption 1 the change of the mutual information caused by introducing NCC x_{ij} can be computed by

$$\Delta_{ij} = i'^{\mathcal{Y}}(x_i) + i'^{\mathcal{Y}}(x_j) + i'^{\mathcal{Y}}(x_{ij}) - i^{\mathcal{Y}}(x_i) - i^{\mathcal{Y}}(x_j). \tag{4.47}$$

where $i'^{\mathcal{Y}}(x)$ is the predictive value function under an updated translation model including x_{ij} in the vocabulary $\mathcal{X}$.

If $\Delta_{ij} > 0$, then $x_{ij} \equiv \langle x_i x_j \rangle$ is a valid NCC for the given application. Assumption 1 allows us to test all NCCs in parallel that satisfy the *mutual exclusion condition*, i.e. a source word $x \in \mathcal{X}$ must not participate in more than one compound. Although this condition is not completely save, because of possible side effects during the learning of a trial translation model, it provides a practically sufficient working hypothesis.

It is still expensive to compute a new trial translation model for each NCC under consideration. Therefore, Melamed suggests an estimation of Δ_{ij} based on the current base translation model. The basic idea assumes that the overall alignment of the word sequences in the parallel text, which is induced by the base translation model, is mostly stable. This is stated by Assumptions 2 and 3,

Assumption 2 *If x_i occurs without x_j in its context $\mathcal{C}$, it will be linked to the same word $y \in \mathcal{Y}$ by the trial translation model as by the base translation model.*

$$i'^{\mathcal{Y}}(x_i) = i^{\mathcal{Y}}(x_i : x_j \notin \mathcal{C}) \tag{4.48}$$

Assumption 3 *If x_{ij} is a valid NCC, then at most one of x_i and x_j will be linked to a target word whenever x_i and x_j co-occur.*

$$Pr(x_{ij}, y) = Pr(x_i : x_j \in \mathcal{C}, y) + Pr(x_j : x_i \in \mathcal{C}', y) \tag{4.49}$$

Note that the probabilities $Pr(x_i : x_j \in \mathcal{C}, y)$ and $Pr(x_j : x_i \in \mathcal{C}', y)$ are based on the counts of links introduced by the base translation model between (x_i, y) and (x_j, y), respectively. Assumption 3 states that the counts should not overlap, i.e. there is no instance of a y in the dataset that contributes to both counts. Even further, we can assume that one of these counts is zero

because the two possible links compete against each other in the translation model (**Assumption 4**).

Taking Assumption 1-4, The final estimate $\hat{\Delta}_{ij}$ can be computed by

$$\hat{\Delta}_{ij} = \hat{\Delta}_{i\to j} + \hat{\Delta}_{i\leftarrow j} \tag{4.50}$$

$$\begin{aligned}\text{where } \hat{\Delta}_{i\to j} = &- i^{\mathcal{Y}}(x_i) \\ &+ \sum_{y\in\mathcal{Y}} Pr(x_i : x_j \notin \mathcal{C}, y) \log \frac{Pr(x_i : x_j \notin \mathcal{C}, y)}{Pr(x_i : x_j \notin \mathcal{C}) Pr(y)} \\ &+ \sum_{y\in\mathcal{Y}} Pr(x_i : x_j \in \mathcal{C}, y) \log \frac{Pr(x_i : x_j \in \mathcal{C}, y)}{Pr(x_i : x_j \in \mathcal{C}) Pr(y)}\end{aligned} \tag{4.51}$$

$$\begin{aligned}\hat{\Delta}_{i\leftarrow j} = &- i^{\mathcal{Y}}(x_j) \\ &+ \sum_{y\in\mathcal{Y}} Pr(x_j : x_i \notin \mathcal{C}', y) \log \frac{Pr(x_j : x_i \notin \mathcal{C}', y)}{Pr(x_j : x_i \notin \mathcal{C}') Pr(y)} \\ &+ \sum_{y\in\mathcal{Y}} Pr(x_j : x_i \in \mathcal{C}', y) \log \frac{Pr(x_j : x_i \in \mathcal{C}', y)}{Pr(x_j : x_i \in \mathcal{C}') Pr(y)}\end{aligned} \tag{4.52}$$

Applications where one half of the parallel dataset is based on a visual vocabulary typically deal with *bag-of-words* translation models. In this case, no language model of the target language is involved and a translation is simply given by translating each word separately by

$$Tr^{\mathcal{Y}}(x) = \operatorname*{argmax}_{y\in\mathcal{Y}} Pr(x, y) \tag{4.53}$$

As a consequence, the predictive value function $i^{\mathcal{Y}}(x)$ (Eq. 4.46) can be simplified to a function $v^{\mathcal{Y}}(x)$ that considers only the maximum probable translation $Tr^{\mathcal{Y}}(x)$ of word x,

$$v^{\mathcal{Y}}(x) = Pr(x, Tr^{\mathcal{Y}}(x)) \log \frac{Pr(x, Tr^{\mathcal{Y}}(x))}{Pr(x) Pr(Tr^{\mathcal{Y}}(x))} \tag{4.54}$$

Then each candidate NCC is judged by

$$\Delta_{ij} = v'^{\mathcal{Y}}(x_i) + v'^{\mathcal{Y}}(x_j) + v'^{\mathcal{Y}}(x_{ij}) - v^{\mathcal{Y}}(x_i) - v^{\mathcal{Y}}(x_j) \tag{4.55}$$

$$\text{where } v'^{\mathcal{Y}}(x_i) = v(x_i : x_j \notin \mathcal{C}) \tag{4.56}$$

$$v'^{\mathcal{Y}}(x_j) = v(x_j : x_i \notin \mathcal{C}') \tag{4.57}$$

$$v'^{\mathcal{Y}}(x_{ij}) = \max[v(x_i : x_j \in \mathcal{C}) v(x_j : x_i \in \mathcal{C}')] \tag{4.58}$$

Eq. 4.58 makes the **Assumption 5** that the most likely translation of the compound x_{ij} is one of the translations of the singletons $Tr^{\mathcal{Y}}(x_i)$ or $Tr^{\mathcal{Y}}(x_j)$ which more frequently occurs in cases where the compound is present. This weaker assumption replaces the stronger Assumptions 3 and 4.

The method of Melamed was suggested in Wachsmuth *et al.* (2003) for driving the grouping process of visual features and was successfully applied in Moringen *et al.* (2008) for the grouping of boundary fragments. In both cases, the notion of context needs to be adapted for treating visual datasets. Melamed distinguishes the right context $\mathcal{C} = RC$ and left context $\mathcal{C}' = LC$ of the instance of a word x. This takes into account the sequential order of words in a text. Wachsmuth *et al.* (2003) define context based on a region-adjacency graph, i.e. all image regions that have a common boundary contour with blob x. Moringen *et al.* (2008) consider all boundary fragments (or compound fragments) that refer to a visual word and are detected in the same image as context. Finally, Jamieson *et al.* (2006, 2007) apply a different objective function but use a similar notion of context by considering the N nearest local image descriptors.

4.5 Summary and conclusion

Parallel datasets are a rich source of information. They provide a coarse grouping of paired collections from different modalities that is the basis for analyzing the statistical dependencies between them. The coarse groupings are given by *situations*. A situation could be defined by a simple pairing of an image and its caption, an observed scene and a spoken utterance, or an action performed and a verbal comment. It is difficult to learn something from a single isolated situation because correspondences between different modalities are not explicitly given. The system has to infer them despite noise, distracting data, and the inherent combinatorics of n-to-m relations.

In this chapter, I have explored algorithms that are learning new concepts by exploiting re-occurring patterns *across situations*. The techniques described in this chapter show that visual models and their corresponding naming can be learned from parallel datasets if sufficient training data is given. However, what can be learnt as a visual model is limited:

- by the feature representation used (blobs, interest points, boundary fragments, etc.),

- by the variations given in the training data (e.g. if a car is alway shown on a street these two concepts are hard to separate).

It is further complicated by synonyms and homonyms on the verbal side and large in-class appearance variations on the visual side. Some of these aspects can only be disambiguated by user-system interaction. However, the exploitation of the statistical properties remains a baseline capability in order to ground concepts on a broad empirical basis.

In learning meaningful visual models from parallel datasets, two different goals need to be distinguished. Approaches that aim at providing topic labels for a complete scene do not need to localize their models in an image. In this area, methods based on co-occurrence statistics are very successful and hierarchical clustering approaches might be the methods of choice if enough evidence can be pooled. Approaches that aim at the localization of scene objects need a different approach that is tighter related to translation models. In this case, only a small portion of the image matches a word item in the corresponding part of the dataset. Features need to be more carefully grouped and spatial relations or configurations become more relevant. The most brittle part of these methods is the initialization phase because the initially linked features need to be descriptive enough in order to bootstrap the model. The attractiveness of models that include an explicit correspondence lies in a possible extension towards the learning of relational structures in both parts of a parallel dataset.

Chapter 5

System Control Strategies

In this chapter, I change from an algorithmic perspective towards a system perspective. A system that is situated in a continuous real environment needs to act. For example, a robot needs to avoid collisions, a surveillance system needs to follow and zoom persons of interest, an assistance system needs to prompt the user in case of relevant events. Besides the external system behavior, there is also the need for controlling the internal processing. Contextual information is relevant for both. In order to achieve a seamless human-machine interaction, the system needs to be aware of the same situational constraints as the user; an active vision system needs to control the camera for finding an appropriate view on the scene; an image interpretation process needs to select the next image operator. The next sections will focus on different control techniques that encode contextual knowledge in some manner.

5.1 Aspects of system control

The control of situated perceptual includes different aspects: the decision what to do next, how to assess goals, how to distribute processing and fuse results, and how to ensure the responsiveness of the system. Control becomes relevant if the system needs to take a sequence of decisions over time in order to achieve a goal or to solve a task in an external *environment*. The task is always related to a *process* that takes place in the environment and that needs to be controlled by the system. A system can be called *situated* if it is able to adapt its goals and control strategy with regard to environmental

factors that are not included in the task definition.

Example: *If a robot needs to find a 'keyboard' in a room, the process that needs to be controlled is the centering of a 'keyboard' object in the view of the camera. There are different actuators that can be used to control this process, e.g. motor commands for moving the robot to a different position and orientation in a room, pan-tilt angles of an active camera mounted on the robot, image operators that try to detect a 'keyboard' object in an image, etc. The environment is given by the room, its lighting conditions, and the global camera parameters that define the current view of the robot. The robotic system would be situated if it would, first, move around to find table tops using a depth sensor, then, locate a monitor using a regular camera, and finally search for a keyboard in front of it. Another example would be that a person tells the robot where to look for a keyboard. In this case, the robot should discard its current strategy and directly look for the keyboard at the specified place.*

The techniques chosen for a particular system depend on the properties of the process that needs to be controlled and the environment where the process is embedded. These can be characterized by the following dimensions,

Observability: The state of the process that needs to be controlled may be fully observable or only partially observable by the control system. In the second case, a process model needs to be updated that explains the observed evidences. This can be seen as an inference task.

Process stochasticity: A partial observability can lead to a stochastic sequence of observations if their relation to internal states is ambiguous. However, the process can also be inherently stochastic rather than deterministic (or the model behind it abstracts from certain dependencies).

Actuation dependency: In many cases, the current decision of a control system will affect all its future decisions. This is typically the case if an irreversible action is performed by the system that changes its relation with the environment. In this case, we would speak of a *sequential* process. Otherwise, each decision can be taken, independently, which characterizes an *episodic* process.

Process dynamics: The control of a process can be seen as a mixed-initiative task that takes place on a time line. *Static* processes keep the initiative on the side of the control system, e.g. interpretation of static images. In *dynamical* processes the environment is continuously changing. Thus, each decision needs to be taken in a specific time corridor until a new event changes the request on the controller. Semi-dynamic environments only change based on an action taken by the control system.

Process continuity: Most real-world environments need to be characterized by *continuous* states and time. However, a process model is frequently abstracted by looking at *discrete time* steps, discrete internal *states*, and discrete *actions* performed by the control system.

Interactivity: In classical control theory, the controller is the single entity that takes decisions that influence the process. In a more general setting this does not need to be the case. Multiple controllers or other active entities can influence the same process, in parallel, having either *cooperative* goals or *competitive* goals. An additional aspect, which comes into play in a multiple entity setting, is the information exchange between these about their mutual internal states and their mutual beliefs about the environmental process.

A special case of an interactive setting is a system that has a dedicated human interaction partner. In the following, we will call him or her the *user* of the system. The user has a specific role with regard to the control strategy because he or she wants to achieve some goal together with the system. Here the system is in a support role and each performance evaluation of the complete system needs to refer to the subjective judgement of the user.

In the following, we will mostly concentrate on environments that are partially observable and inherently stochastic, consist of sequential dynamical processes, are embedded in the continuous real world, and include a dedicated human user of the system. Because of his or her special role, the setup will be treated as a *triadic interaction*[1] (Fig. 5.1). Both the user and the system are able to actively change the environment either by physical actions or virtual

[1] Situations of triadic interaction have been first discussed with regard to the evaluation of complex human machine interfaces and intelligent systems by Bauckhage *et al.* (2002).

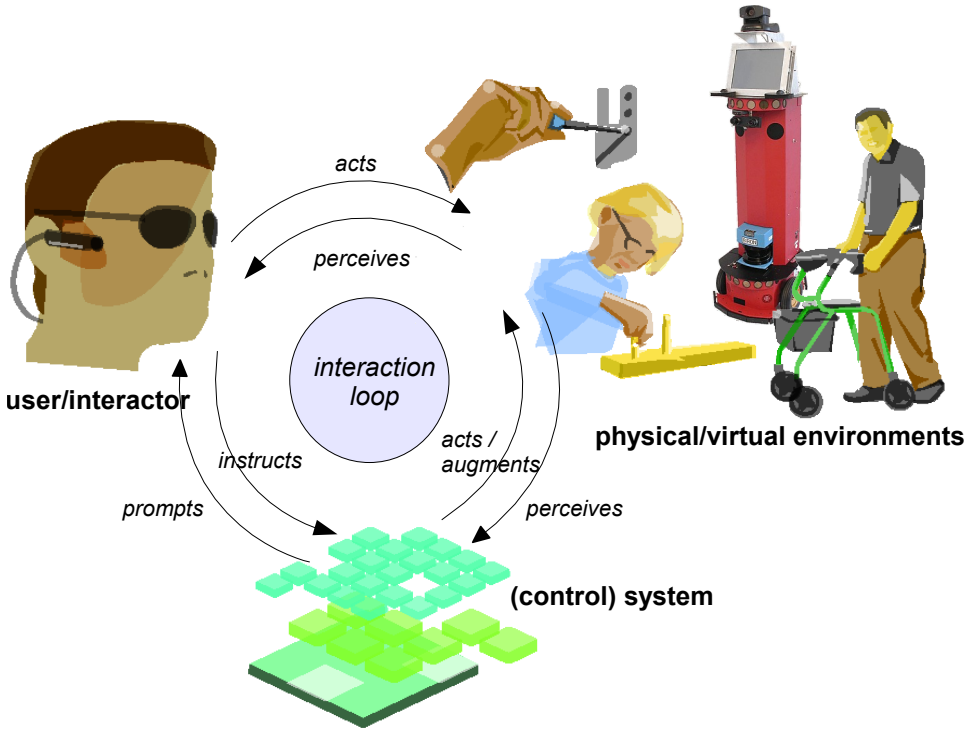


Figure 5.1: Triadic interaction between a user, the system, and the environment

augmentation.[2] At the same time both may be part of the environment which characterizes the communicative setting. Communication could be either direct (e.g. keyboard, pointing devices, etc.) or mediated through the environment (e.g. directed speech, gestures, etc.).

The design, realization, and evaluation of the triadic interaction setting is a complex issue where different research areas are involved (Fig. 5.2). *Human-computer interaction* (HCI) traditionally studies usability aspects of interactive software and hardware devices. It does consider the context of usage but mostly assumes a complete observability of the (virtual) environment which typically is limited to the computer desktop. *Machine perception and robotics* concentrate on interactions between a system and its physical environment. Most work is dedicated to increase the autonomy of a system

[2]The virtual augmentation might need special devices on the perception side of the user (e.g. augmented reality glasses) or on the presentation side of the system (e.g. light projections).

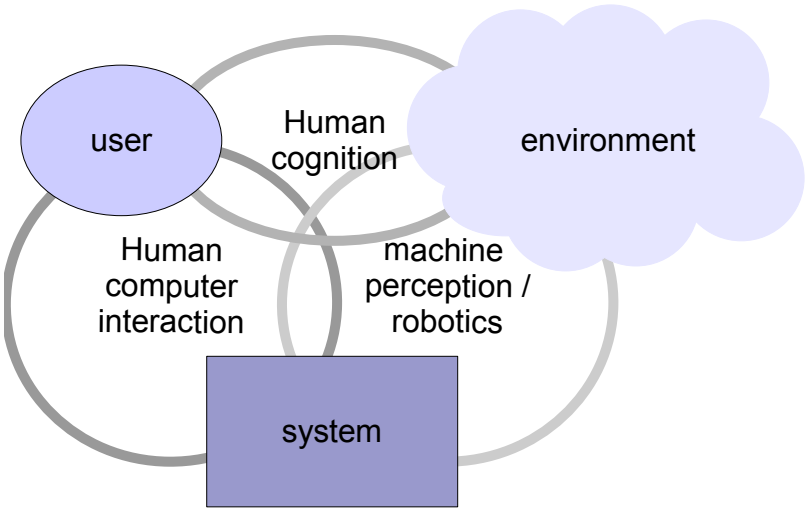


Figure 5.2: Research areas involved in the modeling of a triadic interaction between a user, the system, and the environment

to act in partially observable, stochastic domains. *Human cognition* aims at a deep understanding of the mental processes that underly Human behavior and Human capabilities to act in unconstrained physical environments. This is important for the system's control strategy in so far as the prediction of user intentions will influence the system's acting. Furthermore, the system needs to communicate its own goals in order to provide the user transparent options for his or her acting.

5.1.1 Control theory

The problem of selecting the best action has been dealt with already for a long time. Control theory (Wiener, 1948) treats this problem as the optimization of an objective function over time. Fig. 5.3 depicts the typical setting of a closed-loop controller. The dynamical process that needs to be controlled is characterized by a set of variables. The process can be influenced by means of a control variable which is set by the output of the controller. The observation of an output variable of the process is fed back into the system. An appropriate transform of it is compared to the reference input and the controller decides for an actuation that minimizes the difference between transformed output and reference input, i.e. the error signal. The controller

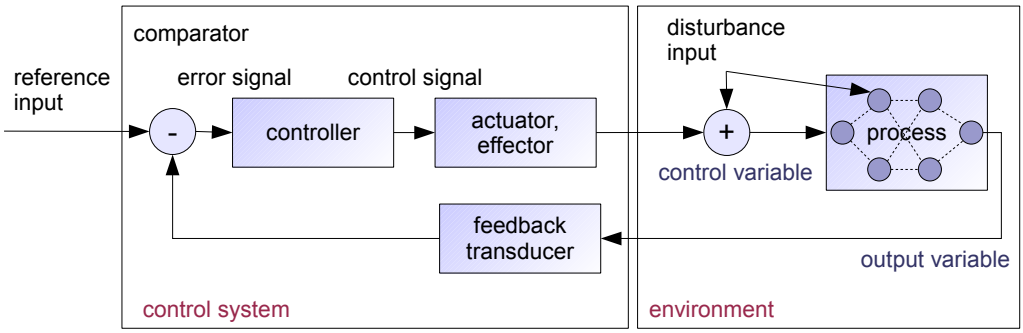


Figure 5.3: Diagram of a closed-loop controller setting

typically includes a (coarse) model of the dynamical process that is able to locally approximate the objective function.

Controlling observable process variables

Control theory is applied to regulate specific quantities of a process, like temperature, pressure, flow rate, etc. The PID-controller is a classical example. The controller sets the value of a control variable $CV(t)$ (e.g. change of heating power, valve position, etc.) which depends on the error $e(t)$ between the output variable and the reference input. In the simplest case, it sums three different terms,

$$CV(t) = P_{out} + I_{out} + D_{out} \tag{5.1}$$

where $P_{out}, I_{out}, D_{out}$ are the contribution of each control component. The proportional term P_{out} proposes a change that is proportional to the error value $e(t)$,

$$P_{out} = K_p e(t), \text{where } K_p \text{ defines the } \textit{proportional gain}. \tag{5.2}$$

With higher values of K_p the system will respond faster to changes, but will continuously over- or under-shoot the reference input and may become unstable. The problems with the oscillatory behavior of the output value can be compensated by adding a D-term proportional to the time-derivative of the error-signal,

$$D_{out} = K_d \frac{de}{dt}, \text{where } K_d \text{ defines the } \textit{derivative gain}. \tag{5.3}$$

The PD controller cannot assure that there will be no steady-state error in the asymptotic behavior. This can be achieved by adding a third term that is proportional to the time-integral over the error-signal,

$$I_{out} = K_i \int_0^t e(\tau) d\tau, \text{where } K_i \text{ defines the integral gain.} \tag{5.4}$$

There are manual as well as automatic methods for tuning the PID-Parameters. The latter are typically based on estimated process models.

Controlling stochastic processes

The PID controller assumes a complete observation of the output variable that is used to compute the error signal. If this is not the case, the error-signal needs to be estimated from partial observations. Thus, the control formalism needs to deal with distributions rather than definitive values. The process is becoming stochastic and the optimal value of the control variable depends on the complete observation history.

However, under the assumptions of a linear-quadratic-Gaussian (LQG) control the *separation principle*[3] guaranties that we can solve the stochastic estimation problem and a (now) deterministic control problem, independently. In case of LQG, the process can be described by a set of linear differential equations with additive Gaussian noise and the objective function is given by a quadratic cost function. Thus, the hidden state $\mathbf{x}_t$ of the process evolves from time step $t-1$ to t by

$$\mathbf{x}_t = \mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{B}_t \mathbf{u}_{t-1} + \mathbf{w}_t \tag{5.5}$$

where $\mathbf{F}_t$ is the state transition model, $\mathbf{B}_t$ is the control-input model applied to control vector $\mathbf{u}_{t-1}$, and $\mathbf{w}_t$ is the process noise with zero mean and covariance $\mathbf{Q}_t$: $\mathbf{w}_t \sim \mathcal{N}(0, \mathbf{Q}_t)$.

An observation $\mathbf{z}_t$ linearly depends on the true process state $\mathbf{x}_t$ with additive Gaussian noise $\mathbf{v}_t$ with zero mean and covariance $\mathbf{R}_t$,

$$\mathbf{z}_t = \mathbf{H}_t \mathbf{x}_t + \mathbf{v}_t, \quad \text{where } \mathbf{v_t} \sim \mathcal{N}(0, \mathbf{R_k}). \tag{5.6}$$

The constraints given for the process model exactly fit the framework of the Kalman filter that can be used to efficiently solve the estimation problem.

[3]The separation principle also applies to other process assumptions, e.g. linear time-invariant systems or quantum systems.

5.1.2 Rational agents

Artificial intelligence (AI) was partially founded in order to overcome limitations of classical control theory which describes each process by a fixed set of continuous variables (Russel & Norvig, 2003). Therefore, AI introduces the more general concept of a *rational agent* that persists over time, operates under autonomous control, perceives its environment, and adapts to change. Instead of providing a numeric reference input, an agent is designed to be capable of adopting externally given goals which is a more abstract concept. The principle of rationality refers to the agent's ability to select the action that achieves the best or best expected outcome. Russel & Norvig (2003) define a rational agent with regard to a percept sequence, i.e. the complete history of what the sensors have ever perceived at each instance in time, and a performance measure, i.e. a success criterion according to what the agent should achieve in its environment.

Definition 3 (Rational agent) *For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has (Russel & Norvig, 2003, p. 36).*

An agent consists of its physical sensor/actuator setting and its agent program. Russel & Norvig distinguish four basic kinds of agent programs.

Simple reflex agents: Agents select actions based on the current percept ignoring the percept history. Programs consist of *condition-action-rules*, also called productions.

Model-based reflex agents: Agents maintain a kind of *internal state* that depends on the percept history. Thus, these agents can deal with partially observable environments. The internal state update uses a coarse model of the world that considers the previous internal state, the current percept, and the own action. The decision about an action is based on the internal state.

Goal-based agents: Agents choose actions that achieve a goal. Thus, these agents need to plan ahead considering future states. They need to reason about their internal state in a planning or search process. In contrast to reflex agents the mapping between internal states and actions is not explicitly represented. The goal can be represented in environmental terms rather than behavior-based terms.

Utility-based agents: Rather than providing binary distinctions between goal states and non-goal states, a *utility function* maps states or state sequences onto real numbers. The agent decides for an action that maximizes the expected utility measurement. Thereby, it can deal with conflicting goals, goal priorities, or partial information about achieving a goal.

Which agent program should be chosen depends on the properties of the environment and the task that needs to be fulfilled. Systems that need to act in real environments even need to combine different types of agent programs and control-types, in many cases.

5.1.3 Coordination of multiple control processes

The overall behavior of a system that interacts with its environment and a possible user needs to fulfill several aspects in parallel. A robot for instance might need to perform a continuous self-localization, track users, objects, and obstacles, avoid collisions, follow some given path, react on speech, etc. Situative control especially refers to the flexibility of a system to deal with multiple or parallel events and its immediate responsiveness on dynamic environmental changes.

This includes several issues besides control, like the consistency of distributed states, the accessibility of information, and the management of system resources. It includes questions about appropriate building blocks for constructing complex system behaviors and appropriate communication patterns for exchanging information between multiple control loops. There are different principled ways to achieve this. Examples of such communication patterns are:

Centralized control (client-server patterns): This is a service-oriented architecture. System components (server) offer different kinds of specialized services that are specifically called by a central control unit (client). The control initiative is based on information or acting needs, i.e. *pull* rather than *push*.

Streaming (publisher-subscriber patterns): Many systems can be built from networks of functional building blocks that continuously receive certain input data and generate a stream of output data. This is an event-notification pattern. A process can register at another process

for a specific event type. The group of registered processes that are notified if the sending process generates new data. There is no central control unit and processes run asynchronously in parallel. The control initiative is characterized by *push* rather than *pull*.

Information-driven coordination (content-based event-notification): The basic event-notification mechanism is limited in so far as the information filter that decides about notification is pre-defined by different groups of events and typically attached to the generating process. Thus, it is difficult to define events on the fly and to filter combined events from different sources. Information-driven coordination abstracts from specific generating processes and notification groups and decides about notification based on the information content that is generated by other processes. Rather than establishing a central control unit, this scheme includes a central view on the data (repository) generated by all system processes.

The different strategies do not completely exclude each other and there are ways to combine different communication patterns in a single system.

5.1.4 User interaction and situation awareness

From the beginning, there has been a debate about the role of AI as a rationalistic or a design approach (Winograd, 2006). The further refers to the vision to model people as cognitive machines that are able to autonomously act in human environments, i.e. the construction of a kind of "super-brain" (as propagated by John McCarthy and others). The latter aims at the vision to realize an intuitive, human-style communication between humans and machines. Rather than replacing the human in the system control loop, systems are *designed* to help or to *augment* the human (as propagated by Douglas Engelbart and others). Thus, the human *user* is embedded in the control loop of the system and takes decisions in cooperation with it. Therefore, both need to be aware of their own and each other's situation.

The term *situation awareness* originally refers to the evaluation of human decision making (Endsley, 1995b). It is the result of a process named *situation assessment* that consists of the *perception* of the relevant situation elements, their *comprehension* in order to get an integrated holistic view of the current situation, and their *projection* in order get predictions about the future behavior of the situation elements. The situation awareness is

influenced by various external factors (e.g. complexity of the task, distraction sources in the environment, workload, design of an interface, etc.) as well as individual factors (e.g. memory capacity, experience, training, etc.) In HCI situation awareness is frequently used as an evaluation methodology. Situation awareness of users can be measured by different techniques like SAGAT (Situation Awareness Global Assessment Technique) (Endsley, 1995a). There, domain experts perform relevant tasks in a system simulator (e.g. flight simulator). The simulation is paused at different points in time and the users are asked questions with regard to their current situation. The percentage of correctly answered questions determines the quality measurement.

With regard to technical systems, the term *context-awareness* has been coined in the area of pervasive and ubiquitous computing. Other related key terms are *ambient intelligence*, *physical* or *haptic computing*, the *internet of things* or *things that think*. They share the common vision to eliminate specific computing devices and, instead, to integrate information processing into everyday environments and objects. As a consequence, location-based services are becoming relevant that are context-sensitive to the identity of nearby people, objects, and recent changes to those objects. In this regard, Dey defines context as "any information that can be used to characterize situations" Dey (2001).

The capturing of situations has a longer tradition in AI, natural language processing (NLP), and computer vision. Schank and Abelson introduced *scripts* for understanding stories (Schank & Abelson, 1977), Minski proposed *frames* that coupled procedural knowledge with the descriptive slot entries describing a situation. Semantic networks (Sagerer & Niemann, 1997; Quillian, 1968) encode information that is relevant in a specific situation in a graph structure. Bobrow (1977) introduces the notion of a *schema* that ties together the relevant information about a concept or an event.

5.2 Production systems

A straight forward method for defining system control strategies is the definition of domain-dependent productions or *condition-action* rules. A production system consists of a set of rules, a working memory, and an inference engine. The system starts at some initial state of the working memory and decides for a response when new information becomes available. This changes

the working memory and the system gets into a new state. This causes a new response and so on.

Forward chaining: In terms of inference, production systems follow the method of *forward chaining*. Given a set of known facts all rules are triggered which premises are satisfied. Then, their conclusions are added to the known facts. This process is continued until some query condition is fulfilled or no more facts can be generated. The complexity of a forward chaining algorithm is mainly determined by the *pattern matching* problem that unifies the premises of rules with suitable facts in the working memory. This problem is known to be NP-hard, though good heuristics are available.

Backward chaining: Forward chaining always involves the risk of concluding irrelevant facts. This can be avoided by backward chaining. Here, the goal is unified with every possible consequence of a rule. The corresponding premises of the rule are added as further subgoals generating a stack of goals to be fulfilled. This is repeated until subgoals match to facts in the working memory. Backward chaining is not always possible. In particular, environments that are only partially observable or stochastic do not allow a deterministic planning process. Thus, we cannot assume that the stack of subgoals is still valid after an action proposed has been executed.

Production systems have been quite successful in the past and some of the early commertial AI applications, like XCON (McDermott, 1982), have been based on this technique. Furthermore, the approach has been widely applied in the field of knowledge-based computer vision systems (Strat & Fischler, 1991; McKeown *et al.*, 1985; Ohta, 1980; Hanson & Riseman, 1987)

5.2.1 Coding context in rules

Context can be explicitly coded in pre-conditions of rules. Strat & Fischler (1991) define context sets that govern the invocation of the system's processing steps. These cover 4 different kinds of criteria: (1) *Global contexts* are attributes of an entire scene like daytime or landscape; (2) *location* characterizes the spatial configuration of a scene like touching the ground, coincidence with other object types; (3) *appearance* of neighboring objects may be similar

like neighboring trees; (4) *functionality* describes the role of an object in a scene like supporting another object or bridging a stream.

Their work builds on the CONDOR system that was initially designed for a hypothetical outdoor robot (Strat, 1993). Its vocabulary consisted of natural objects like trees, bushes, trails, rocks, etc. Instead of individual objects, CONDOR recognized entire contexts that are consistent with its world model. Each image understanding (IU) algorithm A had a precondition for being applied, called *context sets*. A context set is a collection of *context elements* CE_i that each has to be satisfied before action A is performed:

$$L : \{CE_1, CE_2, \ldots, CE_n\} \rightarrow A \tag{5.7}$$

where L is the label or name of the class associated with the context set. An example would be:

$$sky : \{\text{SKY-IS-CLEAR, CAMERA-IS-HORIZONTAL,} \\ \text{RGB-IS-AVAILABLE}\} \rightarrow \text{BLUE-REGIONS}$$

All processing occurs asynchronously and all processes have access to the entire knowledge base. In each step, the system performs all actions A that are associated with a satisfied contest set. If information is missing to judge a context element as true or false, it is treated as unsatisfied. These kinds of rules are used as a unified representation scheme for visual interpretation knowledge. The CONDOR system distinguishes three types of context sets (see also Fig. 5.4):

- Type I: Candidate generation extracts hypotheses from images. Context sets limit the situations in which the operator will be applied.
- Type II: Candidate evaluation/comparison establishes a partial order on candidates. Context sets define the conditions under which the metric applied is valid.
- Type III: Consistency determination/Clique formation generates consistent subsets of hypotheses. Context sets define pre-conditions for applying inconsistency checks.

Context sets are defined for each type and each class of a finite recognition vocabulary. The design of these sets is a critical knowledge engineering task which requires a detailed understanding of the vision procedures employed

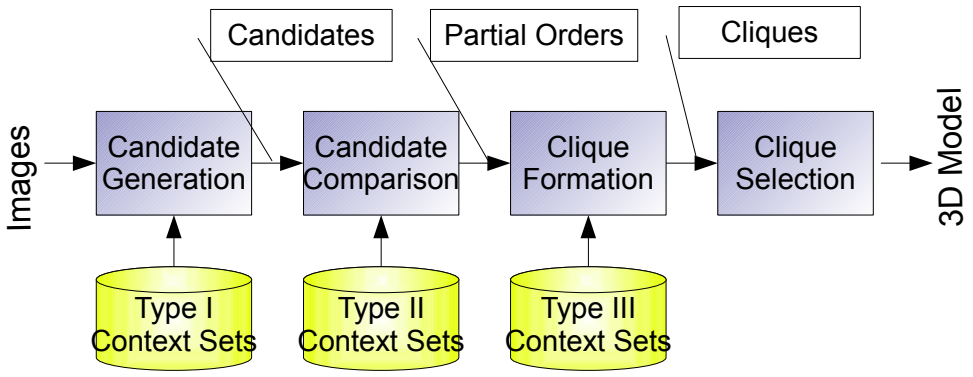


Figure 5.4: CONDOR system by Strat (1993).

in the system. The design philosophy is to keep the single procedures simple, but to provide a large number of them and to achieve robustness by applying multiple operators rather than relying on a single routine.

Forward chaining as well as backward chaining is possible during operation. CONDOR maintains a list of labels that are actively searched for. If the system should search for a tree, for instance, its context set might include a non-validated ground predicate. In this case the 'ground' label would be added to the active list.

In the clique formation process, the system validates the generated hypotheses by searching for consistent candidate sets that explain the largest portion of the image. A set of mutually consistent candidate hypotheses is called *clique*. The sequence of cliques generated is determined by the *candidate comparison* step. The 'Type III' rules trigger *inconsistency procedures* that encode geometric or physical constraints that must be satisfied.

The evaluation of context sets is the core operation of the CONDOR system. This includes the efficient access to scene objects based on local and various semantic properties. The core knowledge structure (CKS) is an object-oriented database, that employs a multi-resolution octree in order to locate objects on any appropriate precision. Similarly, objects are modeled on different levels of detail using a collection of geometric primitives. The recognition vocabulary is organized in a semantic network realizing an abstraction hierarchy.

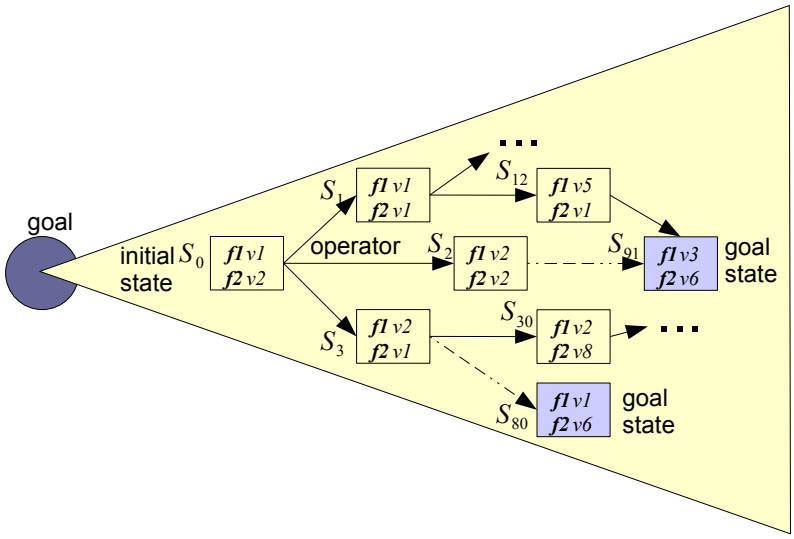


Figure 5.5: Behavior as movement through a problem space. The problem space is partially unfolded in order to search for states that fulfill the goal. States are represented as rectangles and operations as arrows.

Similar approaches that use production rules in order to encode knowledge are the SPAM system by McKeown *et al.* (1985) and a region-based image analysis system proposed by Ohta (1980).

5.2.2 Problem spaces

Production systems face complexity problems if the number of rules or operators becomes very large. In principle, every operator is available at every state and might add new facts to the working memory if these are goal relevant or not. In **Soar** this problem has been tackled by the introduction of multiple *problem spaces* that are organized with regard to goal contexts. Soar[4] (Laird *et al.*, 1987) has been continuously developed since the early 1980s and is still very influential in the area of cognitive architectures. It follows a goal-oriented paradigm of behavior modeling which defines a system behavior as a movement through a *problem space* (Nilsson, 1971; Newell & Simon, 1972). Active problem spaces and their goal contexts are represented in the working memory (WM) of the system. The goal context restricts

[4]Soar is currently available as version 8.

the possible operators that might be applied to a current state. The productions define the knowledge base which is stored in a long-term memory (LTM). Rather than applying productions directly to states, these operate on a problem space by adding or removing elements in it. The control cycle has five phases:

1. Input: New sensory data becomes available in WM,

2. Proposal: Productions fire for interpreting new data (state elaboration), for proposing operators given the current situation/state (operator proposal), and for adding operator preferences (operator comparison),

3. Decision: The operator proposals and preferences are evaluated to select the most appropriate operator. This implements the principle of rationality. In case no operator has been proposed or the operator cannot be decided, an *impasse* is detected and a subgoal with a new problem space is instantiated in WM. These are organized in a stack suspending the current problem space until the subgoal is satisfied.

4. Application: Productions fire to apply the operator to the current state (operator application),

5. Output: In case of agents operating in external environments the output commands are sent to some actuator.

The same control cycle can be used in two different modes: (1) If the system acts in a real environment, operators send motor commands and sensors capture the state changes. (2) If the system just reasons about possible consequences of acting, operators lead to direct state changes.

Productions in the LTM are implicitly organized with regard to problem spaces. Most of the rules code the name of the problem space where they could be applied in their premises. In principle, rules are allowed to work on all slots of a goal context. These are defined as four tuples consisting of

1. a *goal*: the specification of a desired state,

2. a *problem space*: the particular states and operators that might occur relative to a goal,

3. the *current state*: the description of the agent's situation (set of features),

4. a proposed *operator*: the operator that will be applied in the next control cycle.

Knowledge bases of Soar can become quite large if some realistic application needs to be designed. It can be seen as a specialized programming language that also includes a reflexive behavior through the impass detection.

A further influential production system is ACT/ACT-R (Anderson & Lebiere, 1998). It was continuously developed since 1970s. Here productions have an expected cost (time needed) and a probability of success that enable the forward chaining algorithm to follow a more focused search.

5.3 Frame-based systems

Production systems need a central control loop that decides about the next rule to be applied on the content of the global working memory. Productions are the basic organizational units that can independently be added or eliminated. The high degree of modularity comes along with an inflexibility when defining more complex control strategies. There is no notion of local states and the order of an operational sequence needs to be defined indirectly by specifying appropriate rule premises. A strategy to solve this is to define control over larger units that directly couple representations with control knowledge managing their own matching process.

Frames as introduced by Minsky (1975) provide such an instrument. They group different pieces of knowledge into larger chunks that are invoked as a unit. Thereby, frames define a certain means of context or situation. Frames are organized into a number of pairs of *slots* and *fillers* that roughly represent properties and their values. Their verification process matches available features to slots and possibly asks additional questions in order to extract more features or to instantiate additional sub-frames. This process is controlled by auxiliary information stored in the same frame. Walker *et al.* (1988) use frames in order to model object parts and geometric relations. These are arranged in a class hierarchy. The control in such systems is more flexible than in simple rule-based systems, because the order of computation is controlled by accessing the objects' attribute values. This realizes a call by need strategy that invokes active procedures attached to the frames.

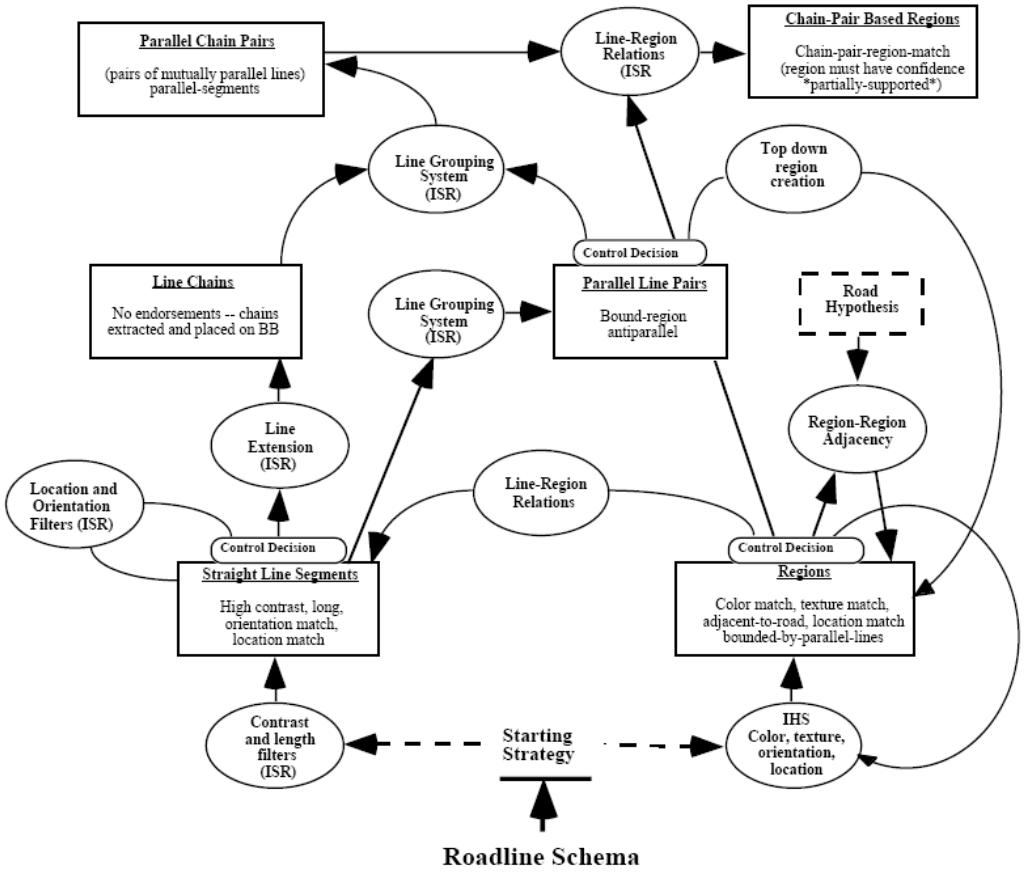


Figure 5.6: A schema related interpretation strategy for finding roadlines in the VISIONS/Schema system. Taken from Draper *et al.* (1996).

5.3.1 Schema theory

Schema theory shares similar concepts with frames. An example is given by the Visions/Schema System (Draper *et al.*, 1989). It explicitly used contextual objects, e.g. "road scene" or "house scene", which grouped contextually related objects. Contextual indexing was used in order to trigger related schemas, e.g. if a 'road' was recognized the schema for recognizing 'telephone poles' was triggered. Fig. 5.6 shows an example a schema for finding roadlines in images. It is formulated as an active process that encapsulates the knowledge, memory, and control procedure for the related object class. The *roadline*-schema has two control threads. One is based on image lines

and the other on image regions. Boxes are local memories, circles represent image operators, and arrows indicate the information flow. Control decisions are based on matching operations that compare items stored in the local memories to some model-database.

The representational formalism of the system is motivated by the schema theory as it was put forward by Arbib (1992). Here, a schema is both a storage of knowledge and the description of a process for applying that knowledge. In this sense, schemas provide a functional decomposition of a system behavior. In an interpretation process multiple instances of a schema may exist, that work like concurrent computing agents that pass messages back and forth in order to achieve a common goal. In principle, schema theory does not need a central and global control instance. Instead, system behaviors emerge from distributed processes of competition and cooperation. Similar ideas have also been proposed in Minski's *Society of Minds* (Minsky, 1988) and in AI research on agent-based approaches (Wooldridge & Jennings, 1995; Shoham, 1999).

5.3.2 Semantic networks

Rule-based approaches as well as schema systems directly model the inference process. In complex scenarios this becomes quite cumbersome because it is difficult to track the consistency of the rule set. Schema systems tackle this problem by defining distributed control units that interact on a coarser level of granularity. A different approach is to take an intensional perspective and to model the domain knowledge rather than the control strategy. Typically, related knowledge can be more efficiently organized and structured in terms of domains rather than control. Semantic networks are a well established formalism to describe explicit domain knowledge (Quillian, 1968; Woods, 1975; Brachman, 1977; Brachman & Schmolze, 1985). A clearly defined semantics of different types of nodes and different types of links leads to the possibility to define domain-independent inference rules.

An example is given by the semantic network language ERNEST (Erlanger Network System). In ERNEST four types of nodes are distinguished.

Concepts: A *concept* is an intensional description of an object, event, or other abstraction. They include general class properties that are independent of any specifically related signal.

Instances: An *instance* is an extensional description. It represents a certain subset of the sensor data that is associated with a specific concept.

The remaining two node types represent an intermediate state of processing. They allow to formulate bottom-up as well as top-down inference strategies.

Modified concepts: a *modified concept* is still abstract but has been constrained by the context of already available instances.

Partial instances: a *partial instance* is associated with a certain subset of the sensor data but misses contextual information for completing the interpretation of it.

Nodes are related by different link types. Besides **specialization** that introduces inheritance to the knowledge base, nodes can be linked to **parts**. If a part cannot be instantiated independent on its own properties and sub-parts, it is context-dependent. For example, a *front wheel* is not well defined without the context of where it is mounted, e.g. a car. A third type are **concrete links** that introduce a hierarchy of representational abstractions, like syntax, semantics, pragmatics, and dialogue in the case of speech understanding systems. Finally, *reference links* can be established, e.g. for pronouns, that refer to a noun phrase.

The domain independent inference rules are based on these notions. The system aims at the more or less complete interpretation of the signal data. It starts with an empty interpretation and successively expands it covering further parts of the signal. Each complete or partial interpretation is given by a network structure that is derived from the knowledge base. The system needs to deal with competing interpretation alternatives as well as with finding efficient operator sequences that minimize computation costs. Therefore, a search process is formulated over network structures. The search space is incrementally explored by applying transformations on network structures that are defined by five different rules.

Rule 1: Generates a *partial instance* from a (modified) concept, if all obligatory, context-independent *parts* or *concretizations* are bound to a partial instance and (if needed) a partial context instance is given.

Rule 2: Generates an *instance* from a partial instance, if all obligatory, context-dependent *parts* or *concretizations* are bound to instances.

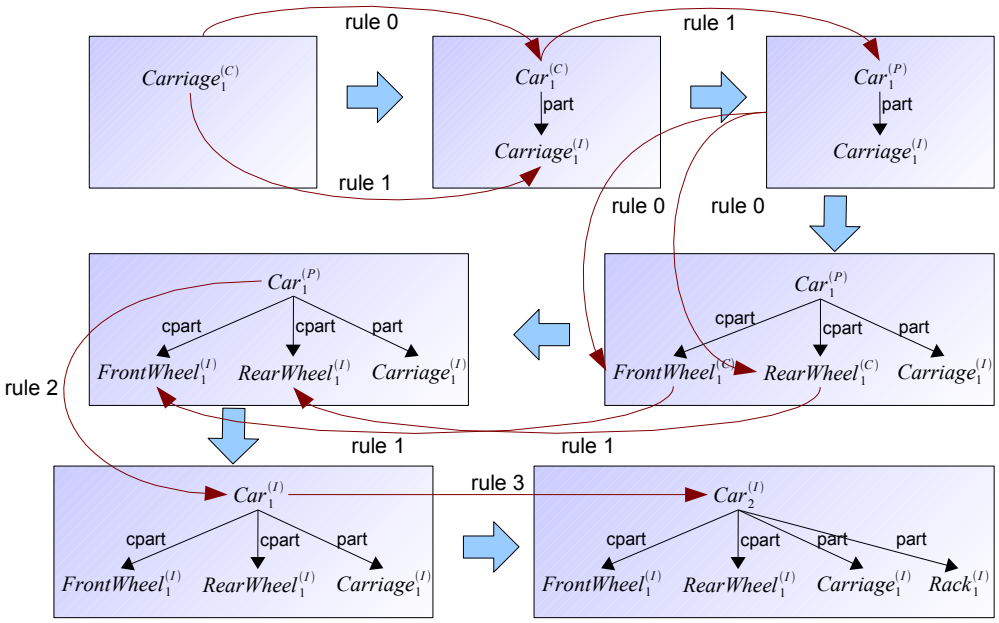


Figure 5.7: Reasoning process in the semantic network language ERNEST.

Rule 3: Expands an *instance*, if an optional part or concretization is bound to an instance.

Rule 4: Generates a *modified concept* from a (previously modified) concept, if an object is bound to a part or concretization of it.

Rule 5: Generates a *modified concept* from a (previously modified) concept, if it is bound to an object as a part or concretization.

The inference process is started by a default **rule 0** that simply generates a concept from a concept class given in the knowledge base. If this concept does not have any parts or concretizations, rule 1 can be applied to drive further inferences. The inference process stops if the interpretation sufficiently covers the signal. Fig. 5.7 gives an example for an inference chain that instantiates a car in an image. With each rule application a set of concept-specific functions are executed for the newly generated node object. This includes feature computation, reference linking, and scoring of features, links, relations, and node. This realizes a call-by-need scheme for feature computations that are only executed in the context of a modification or instantiation of the

corresponding concept.

The linking structure of a semantic network can be further used to propagate contextual information through the network. This is realized by rules 4 and 5 which generate modified concepts. The former generates bottom-up constraints from instantiated parts, the latter restricts parts or concretizations in a top-down manner.

The control strategy decides which rule is applied to which node in which of the alternative interpretations. This can be formulated as a search for an optimal path in a directed graph G. A search node is given by the network of a (partial) interpretation. The costs of an optimal path through search node n is given by two independent terms,

$$f(n) = g(n) + h(n) \tag{5.8}$$

where $g(n)$ are the minimal costs from the start node to n, and $h(n)$ are the minimal costs from n to an end node.

Both terms need to be estimated by scoring functions $g'(n)$ and $h'(n)$. The first term $g'(n)$ is given by the minimal costs of the paths to node n explored so far, and the second term is given by an optimistic estimate $h'(n) \leq h(n)$ of the remaining costs. In this case, the A^* algorithm finds an optimal path in G. The cost functions are domain-dependent and need to be given, externally.

Besides the implicit control strategy formulated by the A^* search, also explicit strategies can be defined. In a data-driven analysis, leave nodes of the part/concrete hierarchy are instantiated first, e.g. based on a segmentation result. Then higher level concepts are processed step by step. In a model-driven analysis, potential goal concepts are initialized by rule 0. Then, these are expanded by generating (modified) concepts of their parts and concretizations until the signal level is reached and instantiation starts. An indepth discussion of different control strategies is given by Kummert (1997).

5.4 Utility-based approaches

Rational agents have been defined as always selecting the action that achieves the best (expected) outcome. If the agent *knows* that an action leads to its goal, it selects it. However, this is typically not a binary criterion. Both, the goal may be vaguely defined and the state may be uncertain. Utility-based approaches combine probability and decision theory in order to capture

both aspects in a homogeneous model. Therefore, a goal is implicitly defined by mapping the possible consequences c of actions a to real numbers that quantify their *utility* $U(c)$ for an agent. The state is characterized by the observational sequence $\mathbf{y}_{1:t}$ experienced so far. The goal of the system is to maximize its expected utility by selecting the most promising action at each time t,

$$U(a) = \sum_{c \in \mathcal{C}} U(c) Pr(c|a, \mathbf{y}_{1:t}), \tag{5.9}$$

where $\mathcal{C}$ is the set of possible consequences and $\mathbf{y}_{1:t}$ is the set of evidences observed by the system until time t.

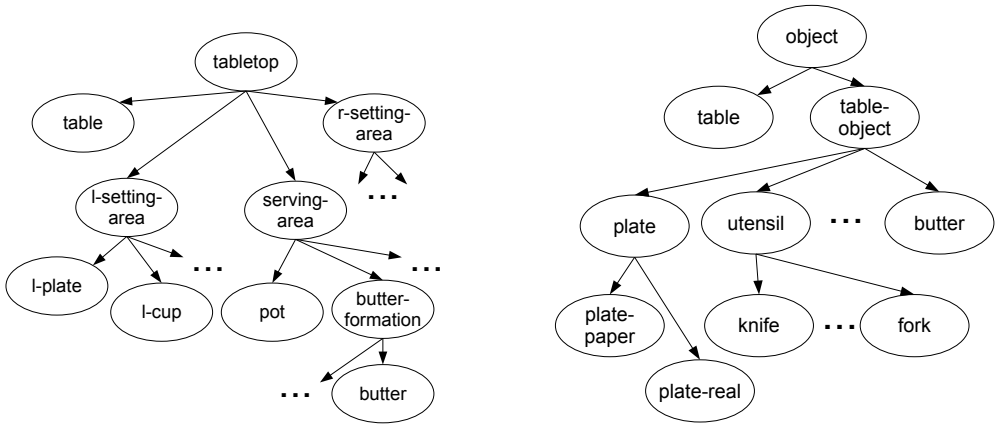
The *maximum-expected utility* (MEU) criterion depends on the set of evidences observed until the time of decision but also needs to consider future decisions that determine the utility of a consequence $U(c)$. This causes several complexity issues for utility-based approaches that need to be considered when designing such systems. State transitions become dependent on an appropriate action selection. And the optimal action selection, in turn, depends on future state changes that determine the utility of the current action decision. Thus, there is a huge search space spanned by a sequence of possible actions that needs to be explored before judging an action decision and a huge number of parameters that need to be estimated. This kind of approach is covered by the theory of *Markov Decision Processes* (MDPs), which assumes complete information, and *Partially Observable Markov Decision Processes* (POMDPs) which are shown to be computationally NP-complete.

5.4.1 Utility-based classification

From the standpoint of active vision each classification task requires the modeling of domain as well as control knowledge. Rather than classifying a single image, the system needs to plan *where to look next* in order to actively explore the scene and gather evidence for judging the classification result. Utility-based approaches ideally extend probabilistic methods for modeling this kind of process. In the following, we will discuss two examples that apply this strategy.

The TEA-1 composite network

The TEA-1 system (Rimey & Brown, 1994) combines probability and decision theory in a two step inference process. Rimey and Brown present an



(a) part-of/area net (b) is-a net

Figure 5.8: Bayesian networks of the TEA-1 system (Rimey & Brown, 1994)

active vision system for classifying *dinner-table* scenes. These consist of hierarchically structured spatial arrangements called *T-world*, e.g. a table has a left setting area that in turn consists of a plate, cup, knife, folk, etc.

The knowledge in the TEA-1 system is represented by four different Bayesian networks. The recursive structure is represented in the *part-of net* (Fig. 5.8(a)). Geometric relations are modeled in the *expected area net* that has the same structure as the *part-of net*. A node in the first network identifies a particular object, and the corresponding node in the *expected area net* identifies the area in the scene in which this object is expected to be located. The domain of the random variables in this network are the positions of an object B on a discrete two-dimensional grid. Conditional probability tables are based on a normalized *relational map* $R_{B|A}(x, y)$ that is scaled and moved by a function f with regard to the anchor object A.

$$P(p_B|p_A) = f(p_B; R_{B|A}, p_A, h_A, w_A),$$
$$\text{where } h_A, w_A \text{ are the height and the width of object } A$$

The classification of each object in the scene is represented by an *is-a net* (Fig. 5.8(b)). It models a taxonomic hierarchy of mutually exclusive subset relationships in the domain. Each node of the *part-of net* is associated with an *is-a net*.

The task-specific knowledge is separated from the domain knowledge described so far. It is encoded in a *task net* (Fig. 5.9) which specifies what

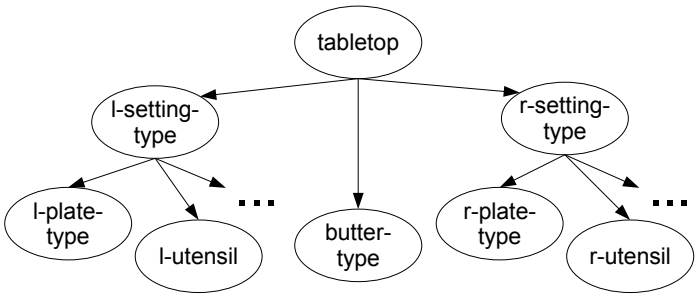


Figure 5.9: Task net of the TEA-1 system of Rimey et al.

objects and object property values are expected for each possible outcome of the task variable.

The four separate Bayesian networks are linked within the *composite net*. The propagation of evidences in this net is realized as follows:

1. Propagate belief in each of the separate nets in the composite net except for the task net.

2. Construct *packages* of *BEL* values from the other nets for transfer to the task net. These packages define the evidences that are attached to the nodes in the task net.

3. Propagate belief in the task net.

TEA-1 combines the contextual belief α about the presence of an object from the part-of net with the detailed classification result β_i where i refers to the node in the is-a net. For the *l-utensil* node the package is:

$$(\alpha\beta_{fork}, \alpha\beta_{knife}, \alpha\beta_{spoon}, 1 - \alpha(\beta_{fork} + \beta_{knife} + \beta_{spoon}))$$
$$\text{where } \alpha = BEL(present) \quad \text{— from the part-of net,}$$
$$\beta_i = BEL(\omega_i), \quad i \in \{fork, knife, spoon\} \quad \text{— from the is-a net.}$$

The next action of the system is selected on the basis of the propagated beliefs of the networks. Either *visual actions* or *camera movement actions* can be performed. Therefore, the problem of *which evidence to get next* is extended by the decision *where to look for evidence*. Each kind of object usually has several actions associated with it. In the table-setting domain, TEA-1 had 21 visual actions related to seven kinds of objects. Any action

has a precondition that has to be fulfilled before the action is executed. For example, the *per-detect-plate* action can be performed if the plate's location is not yet known, if the expected location of the plate is within the visual field for the current camera position, and if the action has not been executed previously. The decision of the best action to be performed is based on the specific costs of an action and the expected effort of an action. The latter is measured by the *expected value of sample information* (*EVSI*). The expected value of the task decision d_i, here if the table setting is *fancy* (d_0) or *not-fancy* (d_1), is defined as

$$EV(d_i) = \sum_{j=0}^{1} V(d_i, t_j)\ P(t_j)$$

$$\text{where } V(d_i, t_j) = \begin{cases} 1000, & \text{if } i = j \\ -1000, & \text{if } i \neq j \end{cases} \quad \text{is a the payoff function.}$$

Here, $P(t_j)$ is the actual belief of the task node if the table setting is *fancy* or *not-fancy*. $EV_0 = \max_i EV(d_i)$ is the payoff value of the optimal decision. The expected payoff value EV_e after performing the action can be defined by means of the piece of evidence e that may be extracted by the action:

$$EV_e = \sum_{k=0}^{n_e} \left[\max_i \left\{ \sum_{j=0}^{1} V(d_i, t_j)\ P(t_j|e_k)] \right\} \right]\ P(e_k)$$

$$\text{where} \quad n_e \text{ is the number of possible values of } e.$$

The expected value of the sample information is then given by the difference between the expected value of the task decision before and after the action is executed:

$$EVSI(e) = EV_e - EV_0$$

The control loop of TEA-1 does not only consider the next action but also sequences of possible actions when deciding which action is to be performed next. The payoff function weights the camera movements against the visual actions and provides a threshold criterion for succeeding.

The probability values of the different Bayesian networks are specified by a human who is familiar with the application domain and task. It turned out that the general behavior of the system is relatively insensitive to variations in the values of the supplied probabilities.

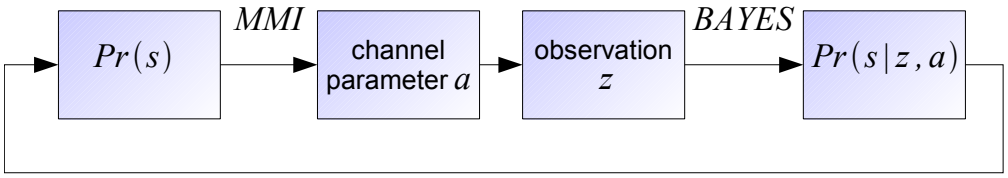


Figure 5.10: Using the maximum mutual information (MMI) in a sequential decision process. The channel parameter a select a new camera viewpoint that minimizes the entropy of the internal state distribution $Pr(s|z,a)$ after observing z.

Using mutual information for control decisions

If the costs of an action do not sufficiently matter, the most appropriate actions can also be defined in terms of the information gain in a specific situation. Denzler & Brown (2002) propose such kind of scheme for an active object recognition task. They control the viewpoint selection of a camera in a sequential decision process (Fig. 5.10) At each time step t we need to select an optimal action a_t leading to an expected observation $Z_t = z_t$ that most precisely predicts the system's state $S_t = s_t$. This is given by maximizing the mutual information between the random variables Z_t and S_t.

$$I(S_t; Z_t|a_t) = H(S_t|a_t) - H(S_t|Z_t, a_t) \tag{5.10}$$

$$= \int_{s_t}\int_{z_t} Pr(s_t, z_t|a_t) \log\left(\frac{Pr(s_t, z_t|a_t}{Pr(s_t|a_t)Pr(z_t|a_t)}\right) ds_t dz_t \tag{5.11}$$

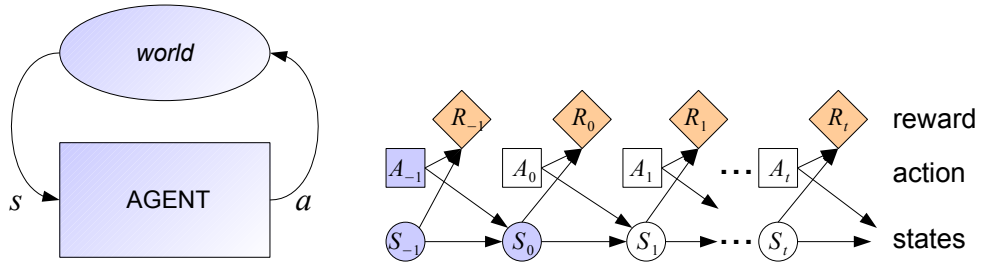
$$= \int_{s_t}\int_{z_t} Pr(s_t|a_t)Pr(z_t|s_t, a_t) \log\left(\frac{Pr(z_t|s_t, a_t)}{Pr(z_t|a_t)}\right) ds_t dz_t \tag{5.12}$$

The double integral can only be solved approximately by Monte Carlo methods (Denzler & Brown, 2002) or need to be discretized. In a sequencial decision process, the *a priori* distribution of the state variable S_t depends on the observations $z_{0:t-1}$ received so far. It can be estimated using the Bayesian filter equations

$$Pr(s_t|z_{0:t-1}, a_t) = \int_{s_{t-1}} Pr(s_{t-1}|z_{0:t-1})Pr(s_t|s_{t-1}, a_t) \tag{5.13}$$

In case of the object recognition task, we have a static rather than a dynamic system. Thus, the equation just simplifies to

$$Pr(s_t|z_{0:t-1}, a_t) = Pr(s_{t-1}|z_{0:t-1}) \tag{5.14}$$



(a) MDP agent (b) MDP graphical model

Figure 5.11: An MDP model assumes a direct access of the environmental state s, but may include stochastic state changes as a result of an action a performed. In the graphical model, the random variable S_0 represents the current state of the system. For deciding on an optimal action $A_0 = a^*$ the system needs to infer the distribution of future states.

The state variable corresponds to the possible object classes $\omega \in \Omega$ and the observations are given by a feature vector f. In order to compute the mutual information at each time step t, we need to estimate the conditional probabilities $Pr(z_t|s_t, a_t) \equiv Pr(f|\omega, a)$ and $Pr(z_t|a_t) \equiv Pr(f|a)$ which can be done from a training set using supervised learning.

5.4.2 Markov Decision Processes

Markov decision processes (MDPs) and related partially observable MDPs (POMDPs) combine probability and decision theory in order to deal with planning and control problems for stochastic processes. In discrete MDPs, the process is characterized by a finite number of states, that are completely accessible for an agent. The agent synchronously interacts with the environment and decides on the next action with regard to a *policy* that maximizes the expected reward of the system. Formally it is defined by a four tuple $\langle \mathcal{S}, \mathcal{A}, T, R \rangle$, where

- $\mathcal{S}$ is a finite set of states,
- $\mathcal{A}$ is a finite set of actions,
- $T : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\mathcal{S})$ is the *state-transition function* mapping an action $a \in \mathcal{A}$ taken in state $s \in \mathcal{S}$ to a probability distribution over world states,

- $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the *reward function* providing the expected immediate reward gained by the agent for taking action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$.

The Markov property restricts the dependency of the expected reward and the next state only to the previous state and the action taken. This is shown by the graphical model in Fig. 5.11. There are also variants of MDPs that deal with continuous state spaces and continuous actions, but this does not change the principal approach.

The policy is a function that maps each state $s \in \mathcal{S}$ to an optimal action $a \in \mathcal{A}$, $\pi : \mathcal{S} \rightarrow \mathcal{A}$. For computing the maximum expected reward the system needs to consider all possible future decisions. This is typically realized by considering a finite-horizon search (limited by time horizon τ) with a discounting factor $0 < \gamma < 1$ that down weights decisions far in the future. Thus, the objective function that needs to be maximized is

$$V_{\pi,\tau}(s) = \mathbb{E}\left[\sum_{t=0}^{\tau-1} \gamma^t r_t\right] \approx \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] \tag{5.15}$$

where r_t is the reward at time step t with regard to a sequence of optimal future decisions. This value function $V_{\pi,\tau}(s)$ can be recursively defined using an induction over time,

$$V_{\pi,1}(s) = R(s, \pi_1(s)) \tag{5.16}$$

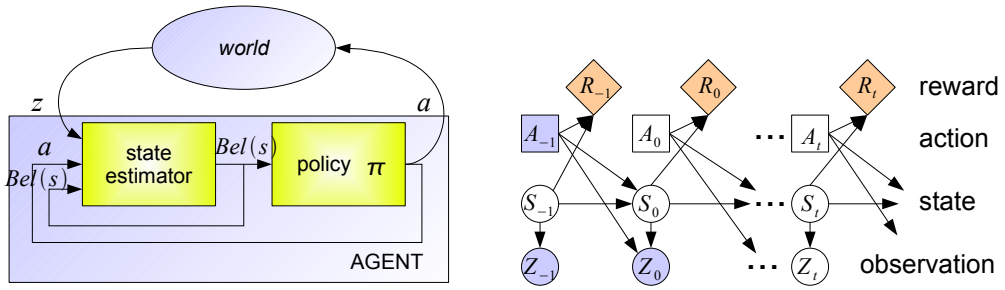
$$V_{\pi,t}(s) = R(s, \pi_t(s)) + \gamma \sum_{s' \in \mathcal{S}} T(s, \pi_t(s), s') V_{\pi,t-1}(s') \tag{5.17}$$

where $R(s, \pi_t(s))$ is the reward of action $\pi_t(s)$ taken at time t in state s
$T(s, \pi_t(s), s')$ is the probability of a state transition from s to s'
given an action $\pi_t(s)$,

The recursive definition introduces a time dependent policy function $\pi_t(s)$, because the time horizon is decreasing with increasing time t. The system might decide differently for the same state if there is only one future step to consider or many. The sequence of optimal policies $\pi_t(s)$ can be computed using the *value iteration algorithm* that exploits the recursive definition of the value function. The optimal decisions are given by

$$\pi_1(s) = \operatorname*{argmax}_{a \in \mathcal{A}} R(s, a) \tag{5.18}$$

$$\pi_t(s) = \operatorname*{argmax}_{a \in \mathcal{A}} R(s, a) + \gamma \sum_{s' \in \mathcal{S}} T(s, a, s') V_{\pi,t-1}(s') \tag{5.19}$$



(a) POMDP agent (b) POMDP graphical model

Figure 5.12: The POMDP model can deal with partially observable processes. Therefore, it decomposes the decision procedure into a *state estimator* considering the last state s, the selected action a, and the observation z, and a *policy computation* based on the estimated state s'. The random variable S_0 of the graphical model represents the current state of the system. Its distribution depends on all previous observations.

An MDP-agent only needs to store the last state in order to decide about its next action. This becomes different if the process is only partially observable. In this case, the agent needs to infer a distribution representing the current belief of the agent about its state. The decision becomes dependent on the complete sequence of actions performed and observations perceived so far. This can be modeled by a POMDP that can be described by a six tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \Omega, O \rangle$:

- $\langle \mathcal{S}, \mathcal{A}, T, R \rangle$ is a Markov decision process,
- Ω is a finite set of observations by which the agent experiences the world,
- $O : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\Omega)$ is the *observation function* mapping an action $a \in \mathcal{A}$ and its resulting state $s' \in \mathcal{S}$ to a distribution of observations.

The decision procedure of a POMDP-agent is decomposed into a *state estimator* and a policy computation. The state estimator updates a belief state that summarizes the previous actions and observations. The policy component maps belief states to actions. Thus, given a correct state estimator, we need to solve a continuous-space MDP for finding the optimal policy. This space grows exponentially with the number of states which makes large problems intractable for computing exact POMDP policies. A summary of

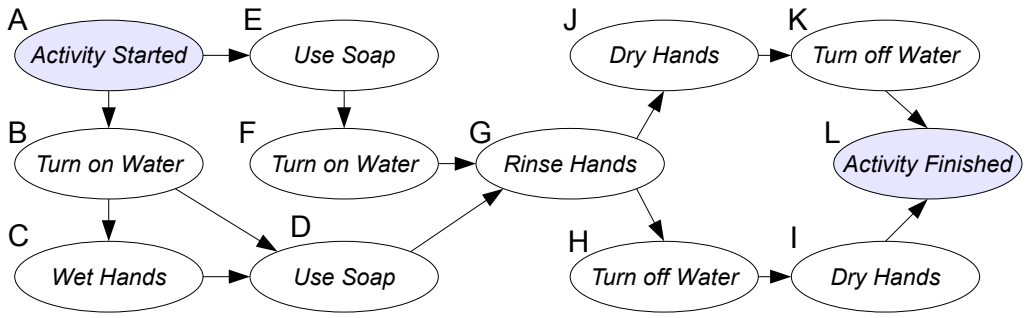


Figure 5.13: Plan graph for the stochastic transition model in a handwashing task.

approximation algorithms for POMDPs is given by Kaelbling *et al.* (1998) or Aberdeen (2003).

An example of a POMDP model is given by Boger *et al.* (2005). Here the policy space is simplified by an underlying plan graph that restricts possible state transitions. In Fig. 5.13 the plan graph is given for a handwashing task. The system aims at assisting people suffering from dementia in *activities of daily living* (ADLs). It actively monitors a user performing a handwashing task and offers task guidance by speech or visual prompting. In order to capture relevant factors for an appropriate system behavior, the state variables of the model include four different aspects:

1. environment variables (hand location, water on/off)

2. activity status variables modeling the maximum and current progress in the ADL plan steps (Fig. 5.13),

3. system behavior variables, like the number of prompts for a plan step,

4. user variables reflecting the responsiveness of the user.

The system has 20 actions providing different specifity levels of prompting, a "null" action, and a "call caregiver" that ends the process. State transitions that lead to a progress or completion of the task are associated with positive rewards while prompts have associated negative rewards. The system has been successfully evaluated in trial experiments with dementia patients at a Toronto hospital (Boger *et al.*, 2005).

5.5 Summary and conclusion

In this chapter, I have discussed different approaches and strategies for controlling the system behavior. Each approach relies on different representations of control knowledge:

- condition-action rules,
- frames and schemas,
- knowledge structures and cost functions,
- probability distributions and reward functions.

Situated computer vision systems have a quite heterogeneous profile so that it will be difficult to base the system's control on a single technique. Following human movements or gaze need different representations than system reactions triggered by specific events or high-level task control. Thus, there will be a bunch of different control paradigms that need to be integrated in a situated computer vision system.

Production rules are attractive because they are easy to specify. However, there needs to be some organizing principles if rule sets become larger and system states become more complex. Here, semantic networks offer many important aspects like inheritance, the grouping of relevant items in frame-like structures, and the possibility to define relations between them. On the negative side, semantic networks are designed for a perceive-reason-act cycle that separates between these three phases. Feedback cycles in human machine interaction need to be more tightly coupled. Perceive, reason, and act are continuous processes running in parallel. As soon as the system starts to act, the human will respond in some way so that, in turn, the system needs to adapt to it. Such kinds of behaviour might be better covered by utility-based approaches like MDPs.

The following chapter, deals with these kinds of heregenous requirements. Rather than looking at the control problem as a monolithic issue, it is treated by a distributed solution. As a consequence, system integration also becomes an important part in system control.

Chapter 6

System Integration

Research in situated computer vision relies on building systems that interact with the environment and with human users. It has been stressed before that there is neither a single inference nor a single representation technique, that will provide a straight forward implementation of the complex system behaviors aimed for. There seems to be an inherent complexity in the task of constructing situated computer vision systems.

In his famous *"no silver bullet"* article, Brooks (1987) distinguishes essential and accidental complexities in software engineering. While accidental complexities are caused by insufficient tools and technologies, essential complexities are inherent to the problem. He concludes that rather than waiting for a tool that will significantly simplify the task, one needs to think about systematic design strategies as an intellectual task.

In the following chapter, I will explore system integration frameworks that solve many of the accidental problems and offer a systematic thinking about the essential problems in the construction of systems that are situated in a real environment.

6.1 Requirements for integrated systems

Perception techniques and system control are not the only issues that need to be solved if we aim for integrated perceptual systems. System integration is a process rather than a single step task. This is also reflected in modern software development approaches like eXtreme Programming (XP) (Beck, 2001) or Unified Process (UP) (Larman, 2005) that propagate an early system inte-

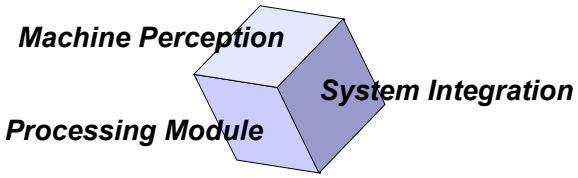


Figure 6.1: Three perspectives on integrated perceptual systems

gration and an iterative expansion and refinement of it during development. Frequent integration cycles determine the pace of the project.

Additionally, complex perceptual systems are rarely built from scratch. Many perceptual sub-tasks require sophisticated algorithmic solutions that involve specific representations, access structures, or control techniques in order to meet performance goals. Therefore, the re-use of independently developed components is indispensable to an effective system engineering approach. Especially, in the scientific community algorithmic development is typically aimed at the general solution of fundamental aspects rather than at a specific solution of an application-relevant task.

This has certain consequences on requirements and demands for system integration methodologies and frameworks. While the functional requirements depend on the specific application, there are several non-functional requirements that generally apply to perceptual interactive systems. These mainly refer to one of three different perspectives (Fig. 6.1):

1. *Perception and interaction:* This is the user and real world perspective. The system is placed in a dynamic real world situation and needs to react on the perceived input in *soft real-time*. This means that the system is able to react to the environment or to a user before the currently processed input becomes outdated. E.g. if the system anticipates a collision with an obstacle in 5sec, it needs to react early enough to avoid the collision. If a user gives a verbal command to the system, it needs to react before the user believes it was missed and he or she starts to give a new one. Perception is typically based on *multiple sensors and modalities*. Additional sensors or modalities should be easy to integrate. However, each sensor as well as the corresponding analysis algorithms have *different timing constraints*. This introduces a couple of challenges for control and information fusion.

2. *Processing modules:* This is a developer perspective. There should be no overhead in contributing new modules and components to the system. On the one hand, this refers to necessary code changes and wrappers needed for common module interfaces or necessary access of external data. On the other hand, it refers to the learning curve of how to program system components. Further overhead may be caused by some component-based integration frameworks that require a formal specifications and performance characterizations in order to enable an overall system control.

3. *System integration:* This is a system engineer perspective. An incremental construction of the system needs to be possible. Distributed processing should be easily supported. A critical question is that of scalability if the number of modules, functionalities, and information exchanges significantly increase. While development of processing modules demands the support of very heterogeneous components and representations, system integration is facilitated by a unified view on its components and exchanged data structures.

There is a large variety of frameworks and tools that support the construction of perceptual systems. These can be roughly grouped into programming libraries, specialized programming languages or language extensions, system infrastructures, and development methodologies. This is not an exclusive classification and many frameworks overlap with more than one group.

Programming libraries: Libraries provide a more or less unified programming interface to a set of implemented techniques and algorithms. Typical examples are *openCV* or *ImageJ* that provide functionality for image processing. However, an all-in-one-lib approach is not a desired solution. The *Image Understanding Environment* (IUE) was originally initiated by DARPA in the 90s as an endeavor towards a common environment for developing computer vision related applications. However, it suffered from a serious "second-system effect" by including too many (mostly irrelevant) features and providing an overloaded class hierarchy.

Language (extensions): System prototyping is frequently supported by providing an abstract specification language for system construction. In this case, the pre-implemented operators and functions are accessed

by an interpreter that defines an abstract machine. A typical example is the *matlab image processing toolbox* or image processing environments like *Khoros* which offers a visual programming language for fast prototyping.

System infrastructures: Infrastructures provide functionality for data/sensor access, inter-process communication, and distributed programming rather than functionality for machine perception. Nevertheless, there are specific requirements for perception systems that justify a specific treatment of this topic. Infrastructures are typically related to development methodologies in that they support different design patterns and architectural principles.

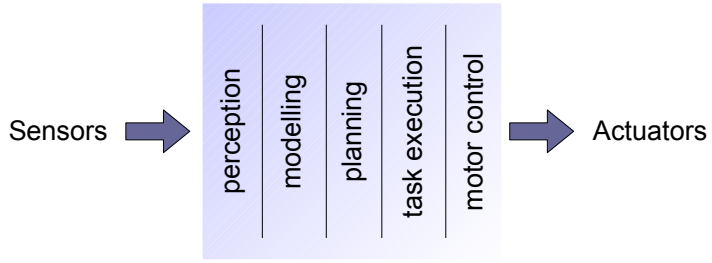
Development methodologies: Methodologies suggest specific principles how to build large perceptual systems. They can be motivated from an engineering as well as biological perspective. How to build systems that robustly perform in real world environments and interact with humans in a seamless manner is still an open problem. Intelligent behavior has been treated in classical AI as a centralized search problem. However, biological evidence as well as lessons learned from system engineering suggest that the overall behavior should emerge from the interplay of smaller components.

The following sections focus on development methodologies and related system infrastructures. There is a large spectrum of different guidelines how to decompose a system into components and how to control their interplay.

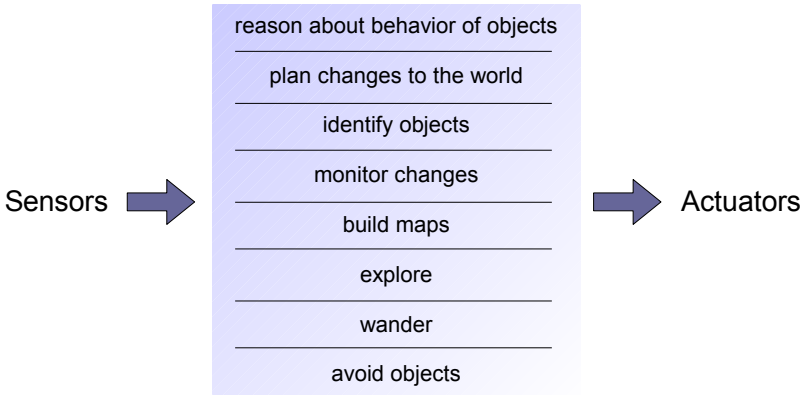
6.2 Behavior modules

The behavior-based approach (Arkin, 1998) has been very influential especially in the robotics community. A behavior is a stimulus-response pattern that contrasts the deliberative planning approach. The system reasons on sensor representations of the external world rather than on internal symbolic representations. This perspective has deep consequences on the construction, modularization, and control of intelligent systems.

The subsumption architecture as put forward by Brooks (1986, 1990) is an early attempt towards behavior-based systems. He provided an alternative methodology to build intelligent systems contrasting classical AI approaches



(a) traditional decomposition into functional modules



(b) decomposition into behaviors

Figure 6.2: Decomposition of intelligent system behavior (after Brooks (1986))

that follow a sequential *perceive - reason - act* control cycle. Fig. 6.2(a) shows the traditional decomposition of a system behavior into functional modules. A central planning unit combines the processing results of the perceptual and modeling functions in order to take goal-directed decisions that are sequenced into elementary motor control commands.

The behavior-based approach takes a different decomposition strategy. The overall behavior emerges from the combination of individual behavior generating modules. These can be defined on different abstraction levels as shown by Fig. 6.2(b). Each behavior directly couples sensor perceptions to actions. These can also include internal sensors that observe other behaviors and internal actuators that change other behaviors.

In the subsumption architecture behaviors are organized in a hierarchy of different abstraction levels (Fig. 6.3). The higher level *subsumes* its lower

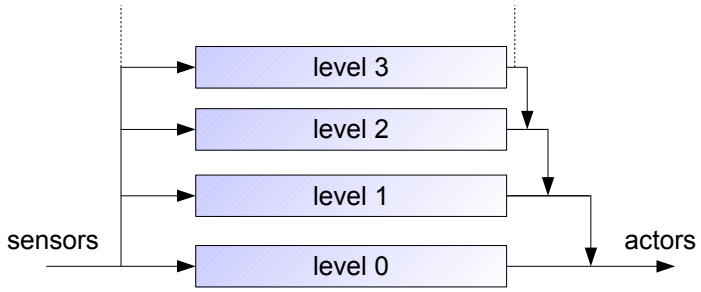


Figure 6.3: Subsumption architecture: higher levels subsume the behavior defined by lower levels.

levels. The *avoid objects* behavior already realizes a limited but complete system. It does not know about any of the higher levels. The *wander* behavior extends the functionality of the system by performing random movements, but it still avoids objects by using the lower level module. In Brook's architecture each behavior consists of several state machines that have defined inputs and outputs. Each state machine has its own local storage structure and there is no global data repository. A behavior can influence a lower level by e.g. substituting the input of a state machine or inhibiting its output. The networks of state machines are fast becoming complex and they should be designed in a strictly bottom-up manner. This makes it somewhat difficult to develop systems in distributed teams and to incorporate externally developed modules. Contexts are implicitly incorporated by the top-down influence of the higher level behaviors.

6.3 Situation controller

A system needs to behave according to specific situations. In behavior-based approaches a situated behavior emerges from the combined output of the hierarchically defined continuous sub-behaviors. There is no explicit modeling of different situations. Taking a different perspective, activity can also be defined as traversing through a network of situations. Each situation needs a reconfiguration of the system's processing modules and control flow. This does apply for the perception of activity as well as for the production of appropriate system responses. The PRIMA system (Crowley & Reignier, 2003) is a good example for this type of system organization that has been applied

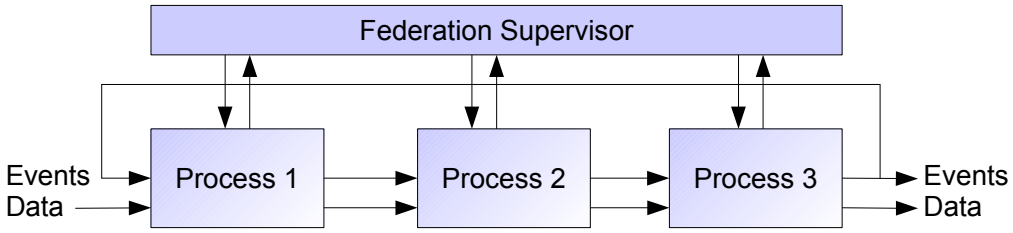


Figure 6.4: A federation of processes in the PRIMA project

in different projects like CAVIAR[1] or CHIL[2].

PRIMA aims at a systematic development of interactive closed environments. They define environments as a "connected volume". An environment is "perceptive" if it is capable of recognizing and describing things, people, and activities within its volume. The environment is "active" if it is capable of changing its internal state. It is "interactive" when it is capable of perceiving, acting, and communicating with its occupants, i.e. responding to humans in a manner that tightly couples perception and action.

PRIMA proposes a conceptual framework for context aware observation of human activity. This is represented as a network of situations. A situation is defined as a configuration of relations (predicate functions) computed over observed entities. Each entity has a role assigned that is an abstract class of *person* or *object* that links to a discriminative recognition test with a symbolic description of functionality. In situation models, roles serve as variables for entities. Changes in relations trigger events that indicate a change in a situation. The whole system is constructed from *federations of processes* (Fig. 6.4). Each federation is composed and controlled by a federation controller ("Federator") in order to predict and observe the situations that describe an activity, and to perform the appropriate actions.

Systems are based on a data-flow architecture based on dynamically assembled federations. A federation is dynamically formed for detecting the entities and relations defining a situation in a specific context. As context changes, the federations are restructured. This enables the system to adapt to a range of environmental conditions and services.

Perceptual processes are composed from a collection of modules controlled

[1] `http://homepages.inf.ed.ac.uk/rbf/CAVIAR/caviar.htm`
[2] http://chil.server.de/

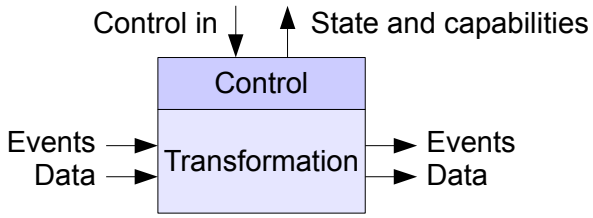


Figure 6.5: Perceptual process in the PRIMA project. A supervisor controls a set of modules. It performs a transformation of a data stream and reacts on as well as generates events.

by a process supervisor (Fig. 6.5). The supervisor is implemented as a multi-language interpreter (Lux, 2003). It allows processes to receive and interpret messages containing scripts and to add new functions using a dynamic loader for pre-compiled libraries. Interprocess communication is provided by a software bus based on the JORAM middleware[3] from ObjectWeb. Modules are formally defined as transformations applied to a certain class of data or event. Modules are executed in cyclic manner by the supervisor according to a process schedule. Besides the transformed data, transformations (modules) also provide an auto-critical report including execution time, confidence in the result, any exceptions encountered. This enables a supervisory controller to adapt parameters for the next call to ensure quality of service.

Supervisory controllers have reflexive capabilities of auto-regulation (processing is monitored and controlled so as to maintain a certain quality of service, e.g. precision vs. time), auto-description (can provide a symbolic description of its capabilities and state, includes a basic set of commands and set of services), and auto-criticism (maintains an estimate of the confidence for its outputs, enable supervisor to dynamically adapt processing). The current state of a process provides its observational variable.

Process controllers can be instructed to give certain priorities to either precision or processing rate. The choice of priority is dictated by a more abstract supervisory controller.

The model provides a means for the dynamic composition of federations of controllers, i.e. observational processes are entities in the system context. A controller can be informed of the possible roles that it may play using a meta-language, such as XML.

[3] `http://joram.objectweb.org/`

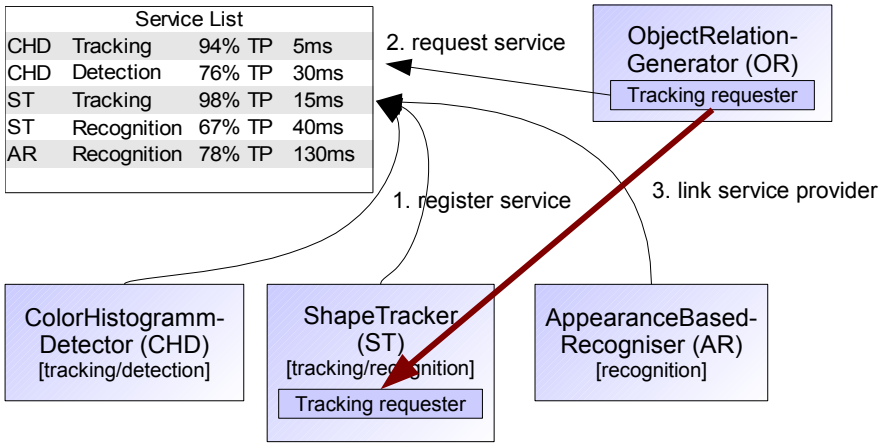


Figure 6.6: Quality of Service (QoS) approach in system integration

6.4 Service-oriented architectures

While the previous frameworks focused on a vertical (behavior modules) or horizontal (network of situation-specific behaviors) decomposition of behaviors, service-oriented architectures provide a functional decomposition. Behavior-control is decided top-down by the specific service request of a client process. Services provide a high degree of decoupling between system components because the interaction between them are dynamically negotiated and linked.

Ponweiser *et al.* (2005) and Vincze *et al.* (2006) present a component-based framework that uses the mechanism of *services* (Fig. 6.6). Each component makes public the services it can perform in the *Service List* (Yellow Pages). Components can check this list for requested services and the communication is automatically established. This mechanism is used for contextual coordination. The processing of the system is task-driven which is implemented by several design guidelines:

- Processing is guided top-down by available task-knowledge that is passed down to components,
- each vision subsystem includes multiple methods for solving a problem that rely on different cues and modalities,
- the framework takes limited resources (number of cameras, processing

power) into account as well as restrictions on the number of persons developing the system.

The scheme is related to multi-agent systems (Wooldridge & Jennings, 1995; Shoham, 1999) which negotiate a configuration that is able to solve a requested task. The framework presented by Vincze et al. is based on software component technology (Szyperski, 1999) that dynamically selects appropriate components and makes it easy to distribute the processing over several computing nodes. Components run asynchronously and are hierarchically organized without an explicit supervisor component. Data consistency is automatically achieved by a data manager.

The communication between components is always established by a *service requester* and a *service provider*. Interfaces are defined as service descriptions which needs to be specified by the component's developer during implementation. Thus, the developer needs to pre-think possible services and uses of a component's functionality. Additionally performance dimensions and constraints permitted need to be specified. The mechanism used for component selection is based on the CORBA trader service[4].

6.5 Data-driven process coordination

Services provide a functional decomposition for goal-driven control strategies. However, many capabilities like the perception of speech, the recognition of gestures, or scene exploration are – to a large degree – data-driven processes. The system directly needs to react on perceptual events. In this case, processes are coordinated through the exchange of data rather than control information. Instead of requesting a service that provides appropriate data items, it directly requests the data stream. An example is given in Fig. 6.7 (Bauckhage *et al.*, 2001). Both the perception of speech as well as images are driven by the capturing process of the signal data. All processing nodes given by the boxes run asynchronously. While the speech processing queue buffers all intermediate results, the image processing queue automatically adapts to real-time by dropping frames. The underlying communication and distributed processing framework is the *Distributed Applications Communication System* (DACS). Its core is based on asynchronous message passing with

[4] Object Management Group (OMG), CORBAservices: Common Object Service Specification, March 1995.

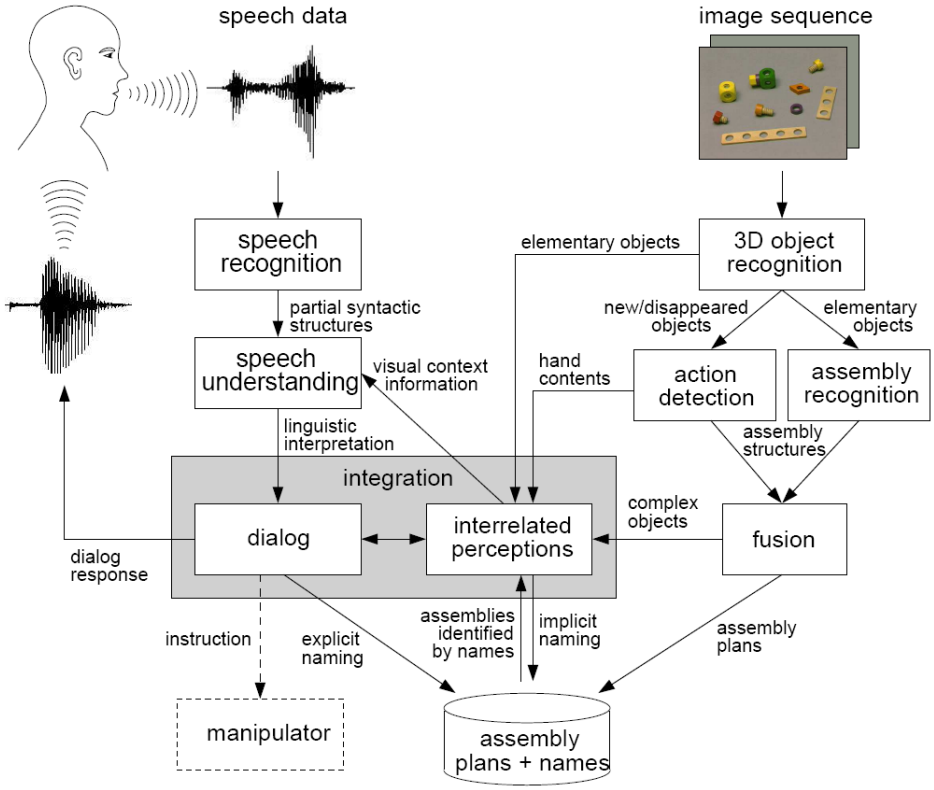


Figure 6.7: Perceptual front-end of a robotic manipulation systems

some extensions for dynamic reconfiguration purposes (Fink *et al.*, 1995). On top of this, DACS offers different communication semantics, like synchronous and asynchronous remote procedure calls (RPCs) and *demand streams*. The latter define a data-driven control mechanism for the receiver side that is triggered by new data generated on the producer side. This is similar to the concept of *event notification*. While this provides an 1:n communication pattern RPCs define a 1:1 communication pattern by establishing a client-server relation. Both are extensively used in the system shown in Fig. 6.7. While the visual as well as the speech processing is based on demand streams, the *interrelated perceptions* module acts as a server for speech understanding and dialog. It provides a working memory that fuses scene related information.

In order to achieve transparent access of relevant data a name server registers streams and RPCs offered by the components. All data is transmitted in a unified reflexive representation language which enables the generic

inspection of data streams by debugging and visualization tools.

Other examples of data-driven coordination are given by Christmas *et al.* (2003) and Christmas *et al.* (2005). Both systems have been developed in the area of automatic annotation of sports video. The former proposes the concept of *semantic cues*. Each cue is realized as a new data stream providing simple concepts detectors like 'grass', 'running track', or 'people crowd'. These can be added without re-designing the whole system. However, the cues need to be synchronized before the final reasoning stage starts. Thus, the system architecture consists of a 4 stage process:

feature detection → cue detection → synchronising → reasoning engine

The latter provides a more flexible framework that includes short-term memory as well as long-term memory. The streaming architecture is automatically built up in a top-down fashion. Once a target module is started it automatically launches other modules providing the input data. This resembles some concepts of service-oriented architectures but with a much lower level of transparency. The short-term memories act as an intermediate buffer for each data stream enabling the re-use of previously computed partial results. The long-term memory stores final annotation results from the short-term memories. This data can be easily accessed and queried using a special browser.

6.6 Blackboards

Blackboards decouple components from fixed or dynamically routed communication channels by mediating all data exchange. All components work on a common data repository which provides a shared working space. Thereby, components are also decoupled in time as the repository stores the data for later access. The blackboard is a passive coordination medium. Thus, the control is realized in a distributed manner in each component. Data exchange between components is realized by a subscription mechanism. A component defines an area of interest in the blackboard and is notified in case of changes. In order to deal with concurrency issues for different modifications proposed to the blackboard, a *supervisor* schedules the access of the components to the repository and decides in case of conflicts (Fig. 6.8). This control mechanism is called *arbitration* and can be based on different principles like order-based execution, priority based scheduling, etc.

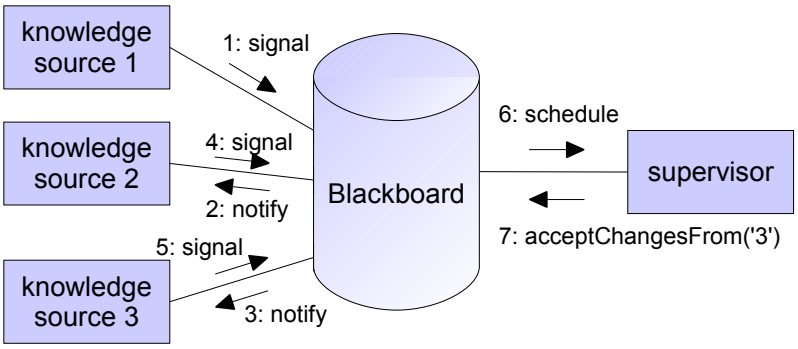


Figure 6.8: The Blackboard pattern decouples data exchanges between components or knowledge sources. These subscribe for an area of interest. If the blackboard receives a signal (1), it notifies subscribing knowledge sources (2,3). If two different knowledge sources respond to a signal (4,5), the supervisor decides which modifications are accepted in which ordering (6,7).

Draper *et al.* (1993) even propose an approach to learning the scheduling algorithm for computer vision tasks. It is based on the VISIONS system which is based on a Blackboard architecture and uses the schema system for supervision control that already has been discussed in Sec. 5.3.1. The Schema Learning System (SLS) applies supervised learning for acquiring object recognition control strategies. These are specialized for a specific object class within a specific context. The learning problem is formalized as the minimization of costs needed to achieve a recognition goal. Given a set of knowledge sources , object models, and training images the SLS learns a recognition graph that optimizes computation time. Then, this graph is used at run time as a scheduling strategy.

6.7 Active memories

Active memories are related to blackboards in that they both use a data repository for data exchange. However, active memories are more than a temporary storage for transient data. They are designed as a medium for knowledge acquisition and offer representations for structuring memory content and establishing relational links between memory items. In this aspect, active memories resemble some ideas of semantic networks which have been discussed in Sec. 5.3.2.

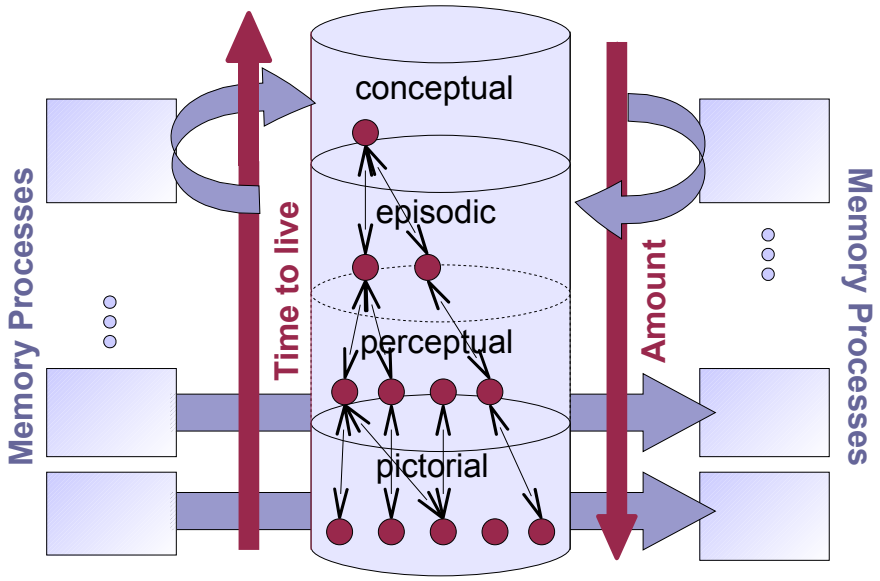


Figure 6.9: Information exchange between the processing modules is mediates by the memory. During interaction the memory builds up representations on different levels of abstraction. The memory content is dynamic. More abstract representation stay longer in the memory. In each abstraction stage the memory works as an information filter.

From a system integration perspective, active memories also differ from blackboards. They follow a more *information-driven coordination* approach. Rather than using a central supervisor to resolve conflicts and schedule knowledge sources attached to the repository, a number of memory processes are defined that maintain the consistency and comprehensiveness of the memory content. Additional memory processes store and retrieve data structures in an asynchronous and parallel manner (Fig. 6.9). A memory server just manages concurrency issues and triggers event notifications. The key elements of an active memory are

- *self-descriptiveness:* The memory does not need to be configured in order to store different kinds of data structures. Changed or new data types can be flexibly stored or mediated at run time. Therefore, the data stored in the memory needs to be *self-descriptive*. A subscribing process declaratively specifies the type of data it is interested in. It does not care about their source or their original format. Thus, a memory

process can dynamically invent new sub- or super-types just by defining an appropriate query. This enables the definition of generic memory processes that keep the memory content dynamic. An example is the *forgetting* process that checks the time-stamp and reliability of memory hypotheses in order to drop irrelevant memory items. Other memory processes can be defined that fuse similar object hypotheses or judge reliability values with regard to contextual information.

- *active coordination of distributed control:* Control is distributed through various memory processes that have their own local control loop. However, the memory takes the coordination part and mediates the information flow between local control loops. A memory process does not need to poll for the current memory state, it is actively notified. A *content-based event notification* is the basic coordination mechanism. On top of this, other memory processes can be defined that purely observe the information flow through the memory and react on it by re-configuring processing modules or re-organizing memory content. In this regard, active coordination also means that the system can adapt through analyzing the content of its own coordination task.

- *gradual treatment of persistence:* The active memory does not structurally distinguish between permanent knowledge and transient data. This enables a more gradual treatment of this issue. Fig. 6.9 shows the conceptual definition of different abstraction layers in the active memory. These can be mapped to different generic types of memory elements.

  - *Pictorial memory* stores *sensor data* that correspond to instantaneous sensor stimuli, like images. The pictorial memory can be used e.g. to extend the field of view by registrating and fusing single camera frames.
  - *Perceptual memory* stores *percepts* that already define a hypothesis of a grouped chunk of features extracted from sensor data. They are still transient but might already be linked to an interpretation. Percepts are candidates for episodic items and can be temporarily stored for learning purposes.
  - *Episodic memory* stores *episodic instances*. These are hypotheses that originate from an interpretation process and exist an ex-

tended period of time.

– *Conceptual memory* stores *concepts* that abstract from a single episodic occurrence. They are typically the result of a learning process and exist a longer period of time.

Rather than being a fixed structure of any active memory instance, the organization into abstraction layers demonstrates the understanding of an active memory as a *self-structuring entity*. Learning is treated as a regular inference process that takes place during memory usage.

The basic principles of *self-descriptive representations*, *content-based event-notification*, and *self-structuring* lead to a *high plasticity* of an active memory. This plasticity is a key element for building highly integrated perceptual systems without large overhead (Wrede *et al.*, 2007; Bauckhage *et al.*, 2008a; Wachsmuth *et al.*, 2007; Fritsch & Wrede, 2007).

Compared to data-driven process coordination – such as demand streams, active memories lead to a further decoupling of components. Demand streams were based on typed event notification channels that were associated with a unique source. A different source that provides the same data type (e.g. object hypotheses) needs to offer a separate event notification channel. Active memories are able to abstract from specific notification channels by mediating the data-flow between system components. The data flow in the system does neither depend on pre-defined connections nor on service negotiation. It is dynamically established based on the content of received and requested data items. Therefore, this strategy is called *information-driven* coordination.

Compared to service-oriented architectures, there is no overhead in providing service descriptions for each component that participates in the system. The philosophy is based on parallelized bottom-up processing rather than top-down scheduling. Instead of requesting services, components request information. This does not directly lead to resource conservative systems. However, it seems to be more robust and it is biologically more plausible to think of resource conservation in terms of *information filtering* rather than service specification (Lütkebohle *et al.*, 2005).

Compared to situation controllers, which coordinate process federations with regard to a specific context, active memories take a different design perspective. Both approaches share common properties, like reflexiveness and auto-regulation. It will be shown in the next section that a similar concept as the federation controller can be realized in an active memory. However, a federation controller directly observes the processing modules while a similar memory process would observe the data flow between the processing modules (mediated by the memory).

Compared to behavior modules, the physical world is not the only interaction space of the system. Behavior-based architectures such as those introduced by Brooks are mainly coordinated through physical world events. Thereby, behaviors can be stated as being grounded. Active memories open up a second virtual interaction space. Each behavior leaves a trace in the memory that can be used for current or later coordination or learning purposes.

6.7.1 The Active Memory infrastructure

Wrede *et al.* (2007) and Fritsch & Wrede (2007) describe a technical realization of the active memory approach. The core technology of active memories is based on the XML-enabled communication framework (XCF). It allows to easily distribute system components over several computing nodes, implements a unified XML-based data exchange protocol, and provides an Active Memory XML server for data management and event-based coordination. Fig. 6.10 provides an overview of the main XCF elements. The memory server is based on the Berkeley DB XML API. Memory processes access data through declarative XPath queries. By this means, a process can either retrieve or subscribe sets of memory items that can be generically specified. For instance, given the XML description of a detected object like

```
<OBJECT>
    <REGION>
        <RECT x="13" y="27" w="80" h="80"/>
    </REGION>
    <CENTER x="32" y="44"/>
    <CLASS>CUP</CLASS>
</OBJECT>
```

sub-elements like *'Regions larger than $w > 50$'* can be easily accessed without caring about outer elements like `<OBJECT>`

```
//REGION[RECT/@w>"50"]                    (6.1)
```

Note that XML-elements are defined in semantic terms like `OBJECT` or `REGION` which supports introspection as well as abstraction possibilities.

The use of XML as a unified representation language serves different purposes. It is a well established and widely accepted method to describe semantic content. It is self-descriptive and – properly applied – supports data abstraction. Meta-information about data-types can be defined in separate XML schema files which can optionally be applied for validating exchanged data structures for debugging purposes. In XCF binary data is natively transmitted similar to the XML-binary Optimized Packaging (XOP) recommendation[5]. The XPath query language provides a powerful and flexible way to access XML data without considering complete data descriptions.

For process distribution and inter process communication the ICE communication engine[6] is used. All data exchanged is specified in XML format. Using the XCF infrastructure, different communication patterns and control strategies can be realized.

- *(a)synchronous Remote Method Invocation* (RMI) is a client-server pattern for 1:1 communication between distributed components.

- *Publisher-subscriber* is a stream-like 1:n communication pattern for data-driven process coordination.

[5] `http://www.w3.org/TR/xop10/`

[6] `http://www.zeroc.com/ice.html`, a new version of XCF is based on Spread `http://www.spread.org`.

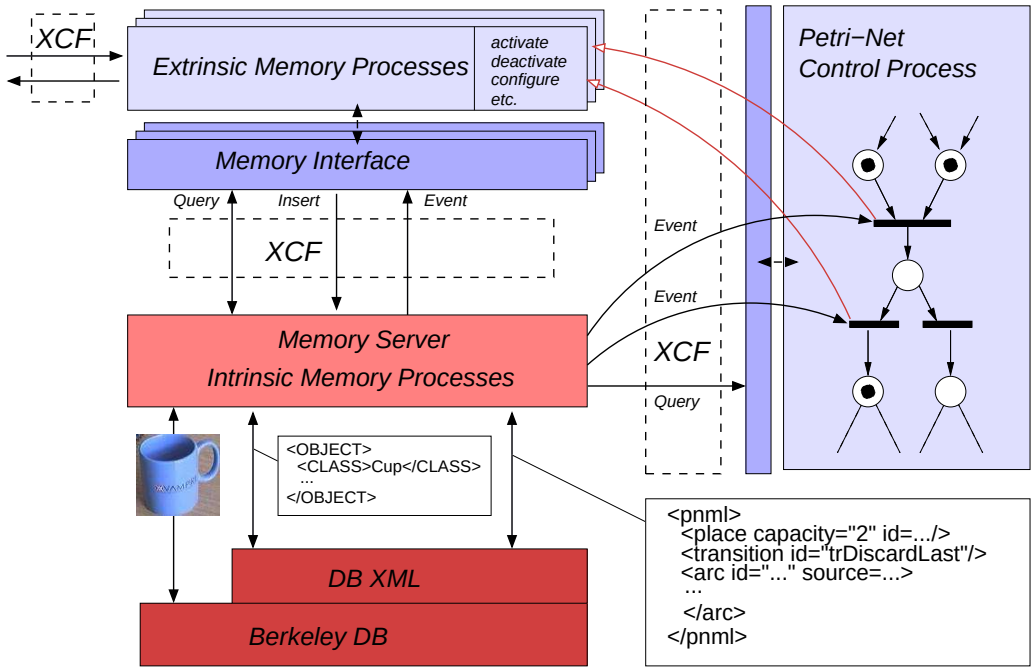


Figure 6.10: Active Memory Infrastructure: Extrinsic memory processes typically have their own computing node and use the memory interface to asynchronously access information stored in the data repository. Intrinsic memory processes can be defined in the computing environment of the memory server for direct and synchronous data access. However, in typical demonstration systems this concept is rarely used. The Petri-Net Control Process can be used to coordinate a federation of memory processes that can be dynamically activated or deactivated using a component interface.

- *Information-driven coordination* is realized through the Active Memory XML Server. Its implementation is based on the native Berkeley DB XML Database[7]. The event system is build on top of it and uses XPath expressions as a subscription language. Whenever a specified *insert, query, update, or delete* action is registered on a data item matching the XPath expression, an event notification is sent to the subscribing component.

- *Petri-net control processes* are the most powerful coordination method that extensively uses the techniques discussed before. Petri-nets extend

[7] `http://www.oracle.com/database/berkeley-db/xml/index.html`

classical state machines by the notion of concurrency. The state of a Petri-net is encoded by the distribution of *tokens* in the *places*. A *transition* fires if all incoming places have enough marks (pre-condition). The Petri-net control processes extend the concept of transitions by linking them to the Active Memory content. Each incoming arc may be associated with an *Active Memory Guard* (AMG) that couples its pre-condition with an active memory event defined by an XPath. This is a two-step process. First, the number of tokens needs to be satisfied. Secondly, the XPath-event needs to be triggered. When the transition fires, a sequence of actions is executed that possibly re-configure specified components.

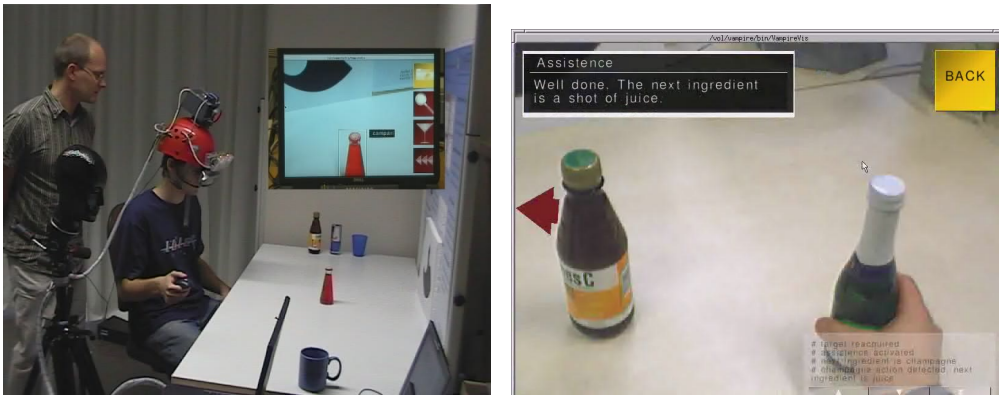
Fig. 6.10 illustrates some of the communication and control patterns. The Petri-Net Control Process *queries* a petri net description that is stored in the memory using the XML-format of the Petri-Net Modeling Language (PNML). Transition arcs are coupled to event notifications and use RMIs for accessing the component interface or other remote methods of a memory process. Memory processes can also directly communicate with other distributed processes using e.g. the XCF publisher-subscriber protocol.

6.7.2 Coordinating memory processes in larger systems

The Active Memory approach has initially been developed in the VAMPIRE project[8] and has been applied in different application systems ranging from interactive AR systems (Siegl *et al.*, 2007) and cognitive task assistance (Wachsmuth *et al.*, 2007) to robotic companions (Fritsch & Wrede, 2007).

The VAMPIRE assistance scenario is shown in Fig. 6.11. The system and the user jointly perform a task of mixing a cocktail drink (Wachsmuth *et al.*, 2007). The VAMPIRE demonstration system actively guides the user through the recipe (Siegl *et al.*, 2004; Wrede *et al.*, 2007). While the user is acting in the scene, the system monitors her/his actions and activities. The head of the user is continuously localized using a hybrid tracking approach (Ribo *et al.*, 2003). It combines a fast CMOS camera with an inertial sensor, both mounted on top of the helmet. The localization approach is based

[8]FP5 IST-2001-34401 Visual Active Memory Processes and Interactive REtrieval (`www.vampire-project.org`).



(a) Scenario (b) User/system view

Figure 6.11: The VAMPIRE system for cognitive assistance: A user is lead by the system through a recipe of a long-drink (Wachsmuth *et al.*, 2007).

on printed artificial landmarks placed in the scene. The user and the system share the same view by applying a video-see-through augmented reality display. The system prompts the user context-related information. During drink mixing it presents the user the next step of the recipe in the visual field and leads her/him to the next ingredient (see red arrow in Fig. 6.11(b)). Additionally, the user can teach new objects and object views to the system in an interactive learning mode (Heidemann *et al.*, 2005).

The VAMPIRE system has mainly been designed in terms of interaction loops that emerge through the memory-mediated coupling of memory processes. Fig. 6.12 gives an example of a simple perception-presentation loop for objects focused by the user. The processing loop is constructed of a visualization and image capturing process (*VIS Image Server*, Wrede *et al.* (2007)), a perception process (*Object Recognition*, Heidemann *et al.* (2005)), and three memory processes (*3D Context, Hypothesis Anchoring, Highlighter*, Hanheide *et al.* (2004); Wachsmuth *et al.* (2005)). The memory processes interact through active memory events. Together with the other two processes they define a continuous feedback loop that is closed by user activities. It is a seamless step to include other recognition processes providing further 3D object percepts or to add contextual reasoning processes that manipulate the reliability of anchored object hypotheses.

A different loop, which is coupled with the first one, is triggered by the user focussing a centered reliable object in the scene (Fig. 6.13). This is formulated as a memory event that is subscribed by the action recogni-

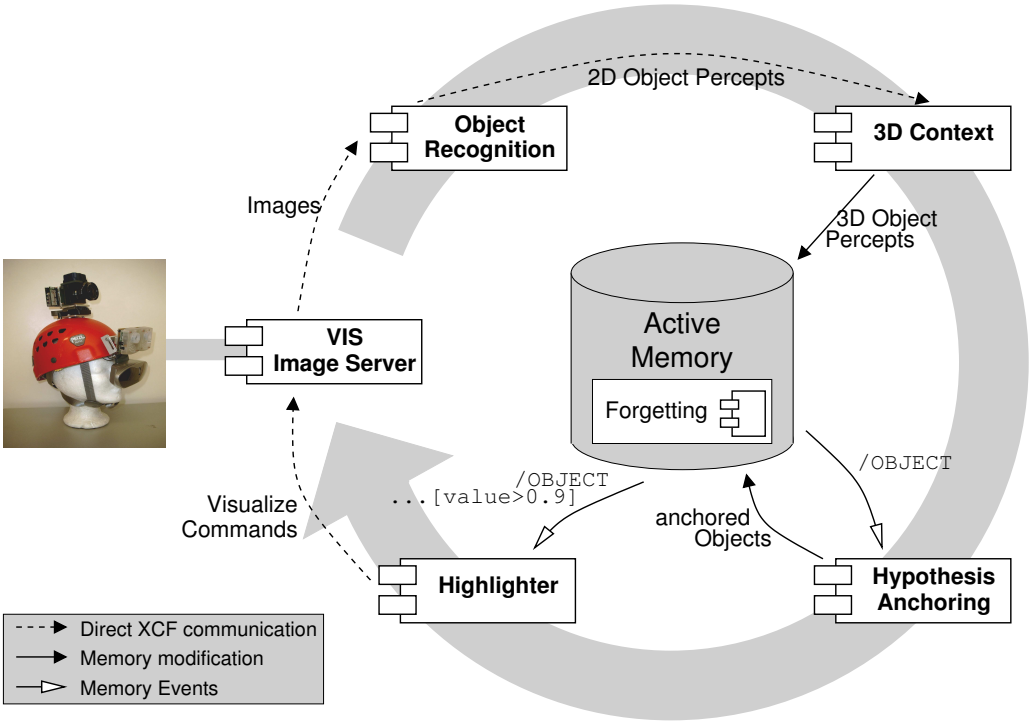


Figure 6.12: Processing loop using an Active Memory, from Hanheide (2006)

tion component. On event triggering, a processing loop is initialized that highlights the tracked object and stores recognized actions in the memory (Hanheide *et al.*, 2006). Due to the high frequency updates of the tracked region the communication to the highlighter is realized by a direct XCF publisher-subscriber stream. Action hypotheses trigger further system activities mediated by the memory, e.g. prompting of the next step in the recipe.

6.8 Summary and conclusion

As already argued before, the construction of complex perceptual systems faces the need to integrate very heterogeneous software components. This contrasts the ambition of integration frameworks to offer more or less homogeneous building blocks for an easy and flexible construction of systems. There are different solutions with regard to this issue:

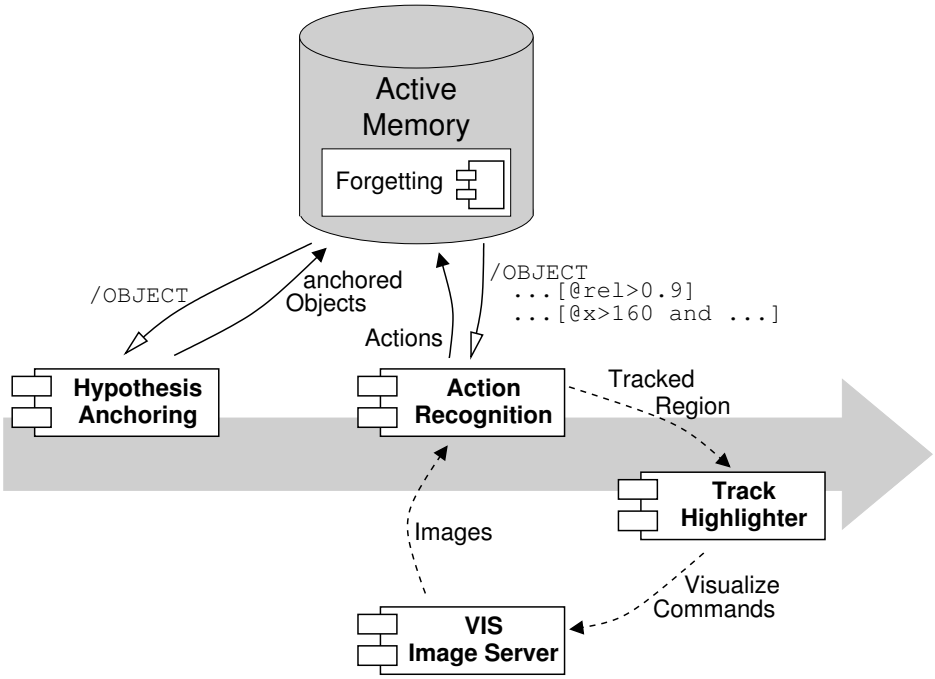


Figure 6.13: Trigger action recognition processes by memory events, from Hanheide (2006)

- In the subsumption architecture, systems were build using the homogeneous building block of a state machine. These were arranged in a layered network with a clear hierarchy. Both restrictions limit the range of possible applications.
- In the federation of processes defined in the PRIMA system, perceptual processes are much more flexible. Nevertheless there is a unique view on such building blocks. This discrepancy is solved by a multi-language interpreter approach. Each processing node is controlled by a script running on such an interpreter. The kind of interpreter is chosen with regard to the specific problem. Thus, the perceptual modules/transformations define abstract machines which communicate through data and event transmission and are (dynamically) *configured/programmed* through interpreter scripts.
- Service-oriented architectures provide a homogeneous view on system

components on the very abstract level of clients and servers. It relies on a description language that formally specifies the services offered by a component. It mainly supports goal-driven control strategies.

- Data-driven approaches describe the integration task in terms of different communication patterns between building blocks. The main concept is based on event-notification.

- Blackboards do not provide any active coordination mechanism. Instead they are used in a passive mode similar to shared memories.

- Active memories propose an information-driven integration approach. Although it offers specific instruments for the control and coordination of system components, it does not impose serious restrictions on system components. This is realized by a high plasticity of the basic technologies used, like XML, content-based event notification, petri-nets, etc.

The frameworks presented solve many of the accidental complexities of system integration and reach a very high level of transparency for implementation issues. However, many essential complexities are left to the designer of such systems. How to decompose the system capabilities? What capabilities are needed for bootstrapping new capabilities? How to design the interaction of the system and the user? What should be stored in the system's memory and what should be forgotten? These questions refer to the understanding of cognition itself, which is still an open debate. Most technical frameworks do not allow to play around with different architectural principles because of a major re-implementation overhead. Although, it has only partially been shown so far the active memory might be a candidate for this endeavor.

Chapter 7

Summary and Outlook

Computer vision is an ill-posed problem. Compared to human vision, even the complete representation of an image in the brain is an illusion. Instead, emprical findings suggest that human vision is a selective process that already starts with the eye fixations. The current situation constrains what is relevant in a scene, where to look in a scene, what should be recognized in a scene, and how to interact with the scene. The perception process is driven by expectations that are inferred from many contextual factors. This is especially true in communicative situations where a human is interacting with a computational device.

Computer vision is becoming more and more important for human-machine interfaces which is driven by miniaturization (smart phones), pervasiveness (intelligent environments), and embodiment (robotic characters). This is also reflected in an increasing number of workshops and conference themes that focus on topics between Computer Vision (CV) and Human Computer Interaction (HCI). There are two lines of research that are related to this. First, there is a large body of work towards *human-centered* computer vision (Turk, 2005; Pantic *et al.*, 2007), i.e. the understanding of human behavior. Secondly, many approaches aim at a kind of *situation awareness* (Crowley *et al.*, 2002; Wachsmuth *et al.*, 2005; Bauckhage *et al.*, 2008b), i.e. the capturing of contextual information that constrains the interpretation task in a specific situation.

Situated computer vision

This book has been dedicated to the second line of research and the direction has been termed *situated computer vision*. Chapter 1 has positioned this term between active vision, context-based vision, and cognitive vision ideas. It also resembles some aspects of situation models. These originate from research in text comprehension, but have been shown to influence an agent's perception as well as action. Understanding can be interpreted as the construction of situation models that in turn provide constraints and expectations for the ongoing perceptual processes. In human machine interaction, this feedback loop needs to be extended towards a human user/communicator. If the system is able to prompt appropriate feedback, a user can react on it by providing additional constraints, scene changes, gestures, or verbal information. These need to be incorporated into the system's situation model that provides the context for understanding.

Thus, context and feedback play a central role in the perceptual process of the system. Chapter 2 and 3 discuss these issues for static scenes as well as human actions. Chapter 4 discusses it from the perspective of learning. There are several re-occurring themes across these techniques:

- *Conceptual hierarchies:* The naming of objects, scenes, and actions may be given on a subordinate, basic, or superordinate level. Currently, most methods focus on a single level that is grounded into perceptual feature spaces. If other levels are considered at all, these are built from elementary parts or basic units. Context could play a much stronger role if grounding would be provided on multiple levels. The hierarchical clustering model discussed in Sec. 4.3.1 explores some steps towards this direction but for a simplified setting.

- *Parametrized models / relational structures / bag-of-features:* In object, scene, and action recognition one has to deal with high in-class variabilities. The techniques discussed deal with different strategies to solve this problem that are partially related to context. Systematic variations of motion models have been dealt with using hidden parameters that model contextual factors like view points. However, parametrized models can only deal with a limited degree of variability. In other cases, relational structures provide a higher degree of invariance but cause complexity issues in matching. A third solution that is frequently found in scene or object recognition completely ignores struc-

ture using a bag-of-features approach. Sometimes even the boundary between object and background is ignored. Bag-of-features works as long as there are a sufficient number of discriminent features. This revealed problematic for cross-situational learning approaches (Chapter 4) where visual models need to be bootstrapped. Here relational structures are essential in order to increase the discriminativity of partial models.

- *Logic vs. probabilistic approaches:* On the one side, logic has the attractiveness to be compositional, i.e. given some basic vocabular of predicates and constants one can build up very complex structures without additional training effort. On the other side, logical expressions quickly become bulky for very simple models. Throughout this book, graphical models have been used in order to discuss various model assumptions. They provide the structure behind the probabilistic world. Probabilistic approaches are mainly attractive because of their fault tolerance. This opens up the chance for contextual cues to provide error corrections. How to combine probabilistic models in a way that allows compositional model construction is still an open question on the computer vision side.

Most systems incorporate only a single aspects of context (like global scene contexts, location, appearance, functionality, tasks, or verbal contexts). This is not sufficient in order reach the goal of a situated computer vision system. There is the need to integrate various aspects of context and to actively shape the context during perception through user-system interaction.

A system perspective

Situatedness becomes relevant under a system perspective where many different situations are faced. In human-robot interaction, augmented reality, or visual assistance scenarios, systems are always running and need to react on perceptual stimuli in an appropriate time scale. As a consequence, the system need to reduce and filter the amount of visual information that needs to be processed. What is relevant depends on the communicative situation between the system and the user. In the VAMPIRE assistance scenario, the user and the system shared the same view. Both interaction partners were able to shape the mutual perception process: the human by turning the head (taking a different view) or by acting in the scene (move some objects); the

system by guiding with displayed arrows, by marking recognized objects, or by displaying icons. The cues will be different for other interaction scenarios, but the same principles of including the human in the system's control loop will apply.

Control strategies should (at least partially) be driven by observable factors. Otherwise, the system will not be able to smoothly interact with a human who continuously provides feedback. This also motivates data- and information-driven integration approaches. System integration can be thought of in terms of parallel (partially) asynchronous interaction loops that are loosely coupled. Being activated in a specific situation, object recognition, self-localization, action recognition, and scene intepretation happen all the time. They provide context for each other and make the perception process less brittle. However, the information communicated to the human interaction partner needs to be filtered with regard its current relevance. The *Active Memory* approach is built exactly on these kinds of principles.

Open challenges

The research direction of situated computer vision is still in its infancy and there are several open issues that will need to be tackled. A few important ones will be named here:

1. How do humans shape each others' perception in a communicative situation?

2. What kind of pre-knowledge and what kind of contextual cues are needed in order to understand complex scenes?

3. How to integrate contextual cues on different scales and abstraction levels?

4. What are appropriate primitives for modeling object, actions, and scenes?

5. How to communicate appropriate feedback that helps the human interaction partner to react on system's perception failures?

Some of these have partially been treated in this book in different variations. However, these probably are not the end of the story.

Currently, many of the human-machine interaction scenarios suffer from the helplessness of the human communication partner if the artificial system does not perceive what they expect. The vision would be that the human interactor would have an intuitive idea – based of the system's feedback – how to react in such situations and to provide the system appropriate contextual cues that lead to a broader base and allow the system to improved its capabilities.

Appendix A

Mathematical details

A.1 Learning translation models

The transformation from Eq. 4.3 to Eq. 4.4 includes the switch of the summation $\sum_{(a_1,\ldots,a_M)}$ and product $\prod_j$ symbols. This step is valid as long as the factors under the product only depend on one of the summation variables a_j:

$$Pr(\mathbf{f}|\mathbf{e}) = \sum_{\mathbf{a}} Pr(M) \prod_{j=1\ldots M} Pr(a_j|L) Pr(f_j|a_j, e_{a_j}) \tag{A.1}$$

We define $Pr(M) \triangleq \epsilon$, $Pr(a_j|L) \triangleq 1/(1+L)$, $\mathbf{a} \triangleq (a_1, \ldots, a_M)$, and write for short $P_{j,a_j} = Pr(f_j|a_j, e_{a_j})$:

$$Pr(\mathbf{f}|\mathbf{e}) = \sum_{(a_1,\ldots,a_M)} \epsilon \prod_{j=1\ldots M} \frac{1}{L+1} P_{j,a_j} \tag{A.2}$$

$$= \frac{\epsilon}{L+1} \sum_{a_1} \sum_{a_2} \cdots \sum_{a_M} \prod_{j=1\ldots M} P_{j,a_j} \tag{A.3}$$

$$= \frac{\epsilon}{L+1} \left\{ \sum_{a_1} P_{1,a_1} \left\{ \sum_{a_2} P_{2,a_2} \cdots \left\{ \sum_{a_M} P_{M,a_M} \right\} \right\} \right\} \tag{A.4}$$

$$= \frac{\epsilon}{L+1} \left\{ \sum_{a_M} P_{M,a_M} \right\} \left\{ \sum_{a_{M-1}} P_{M-1,a_{M-1}} \right\} \cdots \left\{ \sum_{a_1} P_{1,a_1} \right\} \tag{A.5}$$

$$= \frac{\epsilon}{L+1} \prod_{j=1\ldots M} \left\{ \sum_{a_j} P_{j,a_j} \right\} = \frac{\epsilon}{L+1} \prod_{j=1\ldots M} \sum_{a_j} Pr(f_j|a_j, e_{a_j}) \tag{A.6}$$

As a consequence the Eq. 4.7 for the expected count $c(f|e;\mathbf{f},\mathbf{e})$ can be derived as follows:

$$c(f|e;\mathbf{f},\mathbf{e}) = \sum_{\mathbf{a}} Pr(\mathbf{a}|\mathbf{e},\mathbf{f}) \sum_{j=1\ldots M} \delta(f,f_j)\delta(e,e_{a_j}) \tag{A.7}$$

Using Eq. A.6, $Pr(\mathbf{a}|\mathbf{e},\mathbf{f})$ can be expressed by

$$Pr(\mathbf{a}|\mathbf{e},\mathbf{f}) = \frac{Pr(\mathbf{f},\mathbf{a}|\mathbf{e})}{Pr(\mathbf{f}|\mathbf{e})} = \frac{\frac{\epsilon}{L+1}\prod_{j=1\ldots M}\mathbf{t}(f_j|e_{a_j})}{\frac{\epsilon}{L+1}\prod_{j=1\ldots M}\sum_{i=0\ldots L}\mathbf{t}(f_j|e_i)} \tag{A.8}$$

$$= \prod_{j=1\ldots M} \frac{\mathbf{t}(f_j|e_{a_j})}{\sum_{i=0\ldots L}\mathbf{t}(f_j|e_i)} \tag{A.9}$$

This result can be inserted into Eq. A.7 leaving

$$c(f|e;\mathbf{f},\mathbf{e}) = \sum_{a_1,\ldots,a_M} \left\{ \prod_{k=1\ldots M} \frac{\mathbf{t}(f_k|e_{a_k})}{\sum_{l=0\ldots L}\mathbf{t}(f_k|e_l)} \right\} \sum_{j=1\ldots M} \delta(f,f_j)\delta(e,e_{a_j}) \tag{A.10}$$

$$= \sum_{j=1\ldots M} \delta(f,f_j) \sum_{a_1,\ldots,a_M} \left\{ \prod_{k=1\ldots M} \frac{\mathbf{t}(f_k|e_{a_k})}{\sum_{l=0\ldots L}\mathbf{t}(f_k|e_l)} \right\} \delta(e,e_{a_j}) \tag{A.11}$$

For each j only one of the summation variables $(a_1,\ldots,a_M)$ depends on $\delta(e,e_{a_j})$. Thus, for a fixed $\tilde{j}$ we get

$$\sum_{a_{\tilde{j}}} \frac{\mathbf{t}(f_{\tilde{j}}|e_{a_{\tilde{j}}})}{\sum_{l=0\ldots L}\mathbf{t}(f_{\tilde{j}}|e_l)} \delta(e,e_{a_{\tilde{j}}}) \underbrace{\sum_{a_{j'=1\ldots M, j'\neq\tilde{j}}} \left\{ \prod_{k=1\ldots M,k\neq\tilde{j}} \frac{\mathbf{t}(f_k|e_{a_k})}{\sum_{l=0\ldots L}\mathbf{t}(f_k|e_l)} \right\}}_{} \tag{A.12}$$

The second underbraced part sums over all configurations up to 1 leaving a sum over the single variable $a_{\tilde{j}}$ for Eq. A.11

$$c(f|e;\mathbf{f},\mathbf{e}) = \sum_{j=1\ldots M} \delta(f,f_j) \sum_{a_j} \frac{\mathbf{t}(f_j|e_{a_j})}{\sum_{l=0\ldots L}\mathbf{t}(f_j|e_l)} \delta(e,e_{a_j}) \tag{A.13}$$

Because the equation is getting 0 if $f_j \neq f$ or $e_{a_j} \neq e$ we can factor out the fraction for those cases where both word match f and e, respectively.

$$c(f|e;\mathbf{f},\mathbf{e}) = \frac{\mathbf{t}(f|e)}{\sum_{l=0\ldots L}\mathbf{t}(f|e_l)} \sum_{j=1\ldots M} \delta(f,f_j) \sum_{a_j=0\ldots L} \delta(e,e_{a_j}) \tag{A.14}$$

A.2 Mutual information measures

The equations for the mutual information can be easily derived from the basic definition of the mutual information between variables X and Y,

$$I(X;Y) \triangleq H(X) + H(Y) - H(X,Y) \tag{A.15}$$

Substituting $H(X,Y) = H(X|Y) + H(Y)$ we have

$$I(X;Y) = H(X) + H(Y) - H(X|Y) - H(Y) = H(X) - H(X|Y). \tag{A.16}$$

Applying the basic equation of the entropy $H(X) = -\sum_x Pr(x) \log Pr(x)$ to Eq. A.15 gives

$$\begin{aligned}
I(X;Y) = & -\sum_x Pr(x) \log Pr(x) - \sum_y Pr(y) \log Pr(y) && \text{(A.17)}\\
& + \sum_x \sum_y Pr(x,y) \log Pr(x,y) && \text{(A.18)}\\
= & -\sum_x \sum_y Pr(y|x) Pr(x) \log Pr(x) && \text{(A.19)}\\
& - \sum_x \sum_y Pr(x|y) Pr(y) \log Pr(y) && \text{(A.20)}\\
& + \sum_x \sum_y Pr(x,y) \log Pr(x,y) && \text{(A.21)}\\
= & \sum_x \sum_y Pr(x,y) [\log Pr(x,y) - \log Pr(x) - \log Pr(y)] && \text{(A.22)}\\
= & \sum_x \sum_y Pr(x,y) \log \frac{Pr(x,y)}{Pr(x) Pr(y)} && \text{(A.23)}
\end{aligned}$$

Bibliography

Abella, A., & Kender, J. R. 1993 (June). Qualitatively describing objects using spatial propositions. *Pages 33–38 of: Ieee workshop on qualitative vision.*

Aberdeen, Dougles. 2003. *A (revised) survey of approximate methods for solving partially observable markov decision processes.* Tech. rept. National ICT Australia.

Abney, Steven. 1991. Parsing by Chunks. *In:* Berwick, Robert, Abney, Steven, & Tenny, Carol (eds), *Principle-based parsing.* Kluwer.

Abney, Steven. 1996. Partial Parsing via Finite-state Cascades. *In: Proc. of esslli'96 robust parsing workshop.*

Ahmadyfard, A.R., & Kittler, J. 2002. Using relaxation technique for region-based object recognition. *Image and vision computing*, **20**, 769–781.

Aijmer, K. 1996. *Conversational routines in english: Convention and creativity.* London: Longman.

Allen, James F. 1983. Maintaining knowledge about temporal intervals. *Commun. acm*, **26**(11), 832–843.

Aloimonos, Y. 1993. *Active perception.* Hillsdale, NJ: Lawrence Erlbaum.

Aloimonos, Y., Weiss, I., & Bandyopadhyay, A. 1987. Active vision. *Pages 552–573 of: Proc. darpa image understanding workshop.*

Anderson, J. R., & Lebiere, C. 1998. *The atomic components of thought.* Mahwah, NJ: Erlbaum.

Arbib, M. A. 1992. Schema theory. *Pages 1427–1443 of:* Shaprio, S.C. (ed), *The encyclopedia of artificial intelligence*. New York: Wiley and Sons.

Arkin, Ronald C. 1998. *Behavior-based robotics*. MIT Press.

Arulampalam, S., Maskell, S., Gordon, N., & Clapp, T. 2002. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *Ieee transactions of signal processing*, **50**(2), 174–188.

Bajcsy, Ruzena. 1985 (Oct.). Active perception vs. passive perception. *Pages 55–59 of: Proc. 3rd workshop on computer vision: Representation and control*.

Bajcsy, Ruzena. 1988. Active perception. *Proc. of the ieee*, **76**(8), 996–1005.

Ballard, Dana, & Brown, Christopher. 1982. *Computer vision*. Englewood Cliffs, New Jersey: Prentice Hall, Inc.

Ballard, Dana H. 1991. Animate vision. *Artificial intelligence*, **48**(1), 57–86.

Bar, M. 2007. The proactive brain: Using analogies and associations to generate predictions. *Trends in cognitive sciences*, **11**(7), 280–289.

Bar, Moshe, Aminoff, Elissa, Mason, Malia, & Fenske, Mark. 2007. The units of thought. *Hippocampus*, **17**, 420–428.

Barnard, K., & Forsyth, D. 2001. Learning the semantics of words and pictures. *Pages 408–415 vol.2 of: Proc. of eighth ieee international conference on computer vision (iccv)*, vol. 2.

Barnard, K., Duygulu, P., & Forsyth, D. 2001. Clustering art. *Pages II–434–II–441 vol.2 of: Proc. of ieee conf. on computer vision and pattern recognition (cvpr)*, vol. 2.

Barnard, Kobus, Duygulu, Pinar, Guru, Raghavendra, Gabbur, Prasad, & Forsyth, David. 2003. The effects of segmentation and feature choice in a translation model of object recognition. *Pages 675–682 of: Computer vision and pattern recognition*, vol. II.

Bauckhage, C., Fritsch, J., Rohlfing, K.J., Wachsmuth, S., & Sagerer, G. 2002. Evaluating integrated speech- and image understanding. *Pages 9–14 of: Proc. ieee international conference on multimodal interfaces (icmi'02).*

Bauckhage, C., Wachsmuth, S., Hanheide, M., Wrede, S., Sagerer, G., Heidemann, G., & Ritter, H. 2008a. The visual active memory perspective on integrated recognition systems. *Image vision comput.*, **26**(1), 5–14.

Bauckhage, C., Wachsmuth, S., Hanheide, M., Wrede, S., Sagerer, G., Heidemann, G., & Ritter, H. 2008b. The visual active memory perspective on integrated recognition systems. *Image vision comput.*, **26**(1), 5–14.

Bauckhage, Christian, Fink, Gernot A., Kummert, Franz, Lömker, Frank, Sagerer, Gerhard, & Wachsmuth, Sven. 2001. An integrated system for cooperative man-machine interaction. *Pages 328–333 of: Ieee int. symposium on computational intelligence in robotics and automation (cira).*

Beck, Kent. 2001. *Extreme programming explained : embrace change.* Boston, Mass. [u.a.]: Addison-Wesley.

Belongie, Serge, Malik, Jitendra, & Puzicha, Jan. 2002. Shape matching and object recognition using shape contexts. *Ieee transactions on pattern analysis and machine intelligence*, April.

Bernstein, N. A. 1967. *The coordination and regulation of movement.* London: Pergamon Press.

Biederman, I., Mezzanotte, R. J., & Rabinowitz, J. C. 1982. Scene perception: Detecting and judging objects undergoing relational violations. *Cognitive psychology*, **14**, 143–177.

Black, M., & Jepson, A. 1998. Recognizing temporal trajectories using the condensation algorithm. *Pages 16–21 of: Proc. third ieee international conference on automatic face and gesture recognition.*

Blake, A., & Yuille, A. (eds). 1992. *Active vision.* Cambridge, MA: MIT Press.

Bobick, A. F. 1998. Movement, activity, and action: The role of knowledge in the perception of motion. *In: Royal society workshop on knowledge-based vision in man and machine.*

Bobrow, D. 1977. An overview of krl. *Cognitive science*, **1**(1).

Boger, Jen, Poupart, Pascal, Hoey, Jesse, Boutilier, Craig, Fernie, Geoff, & Mihailidis, Alex. 2005 (July). A decision-theoretic approach to task assistance for persons with dementia. *In: Proc. of int. joint conf. on artificial intelligence (ijcai).*

Borgefors, G. 1988. Hierarchical chamfer matching: A parametric edge matching algorithm. *Ieee transactions on pattern analysis and machine intelligience*, **10**, 849–865.

Brachman, R. J. 1977. *What's in a concept: Structural foundations for semantic networks.* Cambridge, MA: Bolt, Beranek, and Newman.

Brachman, R. J., & Schmolze, J. G. 1985. An overview of the kl-one knowledge representation language. *Cognitive science*, **9**, 171–216.

Brill, Eric. 1994. Some advances in transformation-based part of speech tagging. *Pages 722–727 of: Proceedings of the 12th national conference on artificial intelli gence*, vol. 1. Menlo Park, CA, USA: AAAI Press.

Brooks, F. P. 1987. No silver bullet - essence and accident in software engineering. *Computer*, **20**(4), 10–19.

Brooks, Rodney A. 1986. A robust layered control system for a mobile robot. *Ieee journal of robotics and automation*, **RA-2**(April), 14–23.

Brooks, Rodney A. 1990. Elephants don't play chess. *Robotics and autonomous systems*, **6**.

Brown, Peter F., Della Pietra, Vincent J., Della Pietra, Stephen A., & Mercer, Robert L. 1993. The mathematics of statistical machine translation: parameter estimation. *Comput. linguist.*, **19**(2), 263–311.

Buxton, H., & Neumann, B. 1996 (Sept.). *Visual interpretation and understanding.* Tech. rept. EC Vision - European Research Network for Cognitive Computer Vision Systems.

Carneiro, G., Chan, A. B., Moreno, P. J., & Vasconcelos, N. 2007. Supervised learning of semantic classes for image annotation and retrieval. *Ieee trans pattern anal mach intell*, **29**(3), 394–410.

Carson, Chad, Thomas, Megan, Belongie, Serge, Hellerstein, Joseph M., & Malik, Jitendra. 1999. Blobworld: A system for region-based image indexing and retrieval. *In: Third international conference on visual information systems*. Springer.

Chen, S., & Goodman, J. 1996. An empirical study of smoothing techniques for language modeling. *Pages 310–318 of: Proc. of the 34th annual meeting of the acl*.

Christmas, W., Kostin, A., Yan, F., Kolonias, I., & Kittler, J. 2005. A system for the automatic annotation of tennis matches. *In: Fourth international workshop on content-based multimedia indexing*.

Christmas, W.J., Kittler, J., & Petrou, M. 1995. Structural matching in computer vision using probabilistic relaxation. *Ieee transactions on pattern analysis and machine intelligence*, 749–764.

Christmas, W.J., Jaser, E., Messer, K., & Kittler, J.V. 2003. A multimedia system architecture for automatic annotation of sports videos. *Pages 513–522 of: Computer vision systems*. Lecture Notes in Computer Science, vol. 2626. Berlin/Heidelberg: Springer.

Chung, F. 1996. *Spectral graph theory*. CBMS series, vol. 92. American Mathematical Society.

Crandall, David, & Huttenlocher, Daniel. 2006. Weakly supervised learning of part-based spatial models for visual object recognition. *Pages 16–29 of: Proc. of european conf. on computer vision (eccv)*.

Crowley, J. 1995 (February). What is active vision? *In: Workshop on active vision hardware (slide-book)*.

Crowley, J. L., Coutaz, J., Rey, G., & Reignier, P. 2002 (September). Perceptual components for context aware computing. *In: Ubicomp 2002, international conference on ubiquitous computing*.

Crowley, James L., & Christensen, Henrik I. (eds). 1995. *Vision as process*. Springer.

Crowley, James L., & Reignier, Patrick. 2003. Dynamic composition of process federations for context aware perception of human activity. *International conference on integration of knowledge intensive multi-agent systems*, **KIMAS'03**.

de Finetti, B. 1990. *Theory of probability, vol. 1-2*. Chichester: John Wiley & Sons Ltd.

Deerwester, S., Dumais, S. T., Landauer, T. K., Furnas, G. W., & Harshman, R. A. 1991. Indexing by latent semantic analysis. *Journal of the american society for information science*, **41**(6), 391–407.

Demirci, M. Fatih, Shokoufandeh, Ali, Keselman, Yakov, Bretzner, Lars, & Dickinson, Sven. 2006. Object recognition as many-to-many feature matching. *Int. j. comput. vision*, **69**(2), 203–222.

Dennett, D. C. 1996. *Kinds of minds: Toward an understanding of consciousness*. New York: Basic Books, Inc.

Denzler, J., & Brown, C. M. 2002. Information theoretic sensor data selection for active object recognition and state estimation. *Ieee transactions on pattern analysis and machine intelligence*, **24**(2), 145–157.

Dey, A. K. 2001. Understanding and using context. *Personal and ubiquitous computing*, **5**(1), 4–7.

Dickinson, Sven J., Pentland, Alex P., & Rosenfeld, Azriel. 1992. 3-D Shape Recovery Using Distributed Aspect Matching. *Ieee trans. on pattern analysis and machine intelligence*, **14**(2), 174–198.

Draper, B., Collins, Robert, Brolio, J., Hanson, A., & Riseman, E. 1989. The schema system. *The international journal of computer vision*, **2**(3), 209–250.

Draper, B., Hanson, A., & Riseman, E. 1993. Learning blackboard-based scheduling algorithms for computer vision. *Int. journal of pattern recognition and artificial intelligence*, **7**(2), 309–328.

Draper, B., Hanson, A., & Riseman, E. 1996. Knowledge-directed vision: Control, learning and integration. *Proceedings of the ieee*, **84**(11), 1625–1637.

Duda, R., & Hart, P. 1973. *Pattern classification and scene analysis.* New York: Wiley.

Duygulu, Pinar, Barnard, Kobus, de Freitas, João F. G., & Forsyth, David A. 2002. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. *Pages 97–112 of: Eccv (4).*

Endsley, M. R. 1995a. Measurement of situation awareness in dynamic systems. *Human factors*, **37**(1), 65–84.

Endsley, M. R. 1995b. Toward a theory of situation awareness in dynamic systems. *Human factors*, **37**(1), 32–64.

Ericsson, K. A., & Kintsch, W. 1995. Long-term working memory. *Psychological review*, **102**, 211–245.

Eshera, M., & Fu, K. 1986. An image understanding system using attributed symbolic representation and inexact graph-matching. *Journal of the association for computing machinery*, 604–618.

Fink, Gernot A., Jungclaus, Nils, Ritter, Helge, & Sagerer, Gerhard. 1995. A communication framework for heterogenous distributed pattern analysis. *Pages 881–890 of: Proc. ieee int. conf. on algorithms and architectures for parallel processing (icapp)*, vol. 2.

Flickner, Myron, Sawhney, Harpreet, Niblack, Wayne, Ashley, Jonathan, Huang, Qian, Dom, Byron, Gorkani, Monika, Hafner, Jim, Lee, Denis, Petkovic, Dragutin, Steele, David, & Yanker, Peter. 1995. Query by image and video content: The qbic system. *Computer*, **28**(9), 23–32.

Friedman, A. 1979. Framing pictures: The role of knowledge in automatized encoding and memory for gist. *Journal of experimental psychology: General*, **108**, 316–355.

Friedman, J., Hastie, T., & Tibshirani, R. 2000. Additive logistic regression: a statistical view of boosting. *Annals of statistics*, **28**(2), 337–374.

Fritsch, J., Hofemann, N., & Sagerer, G. 2004. Combining sensory and symbolic data for manipulative gesture recognition. *Pages 930–933 of: Pattern recognition, 2004. icpr 2004. proceedings of the 17th international conference on pattern recognition*, vol. 3.

Fritsch, Jannik, & Wrede, Sebastian. 2007. An integration framework for developing interactive robots. Springer Tracts in Advanced Robotics, vol. 30. Berlin: Springer.

Geman, S., & Geman, D. 1984. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Ieee trans. pattern anal. mach. intell*, **6**, 721–741.

Gernsbacher, M.A. 1990. *Language comprehension as structure building.* Hillsdale, NJ: Erlbaum.

Gibson, J. J. 1950. *The perception of the visual world.* Boston, MA: Houghton Mifflin.

Griffiths, A., Luckhurst, H. C., & Willett, P. 1986. Using inter-document similarity information in document retrieval systems. *Journal of the american society for information science*, **37**(3), 3–11.

Hanheide, M., Bauckhage, C., , & Sagerer, G. 2004. Memory consistency validation in a cognitive vision system. *Pages 459–462 of: Int. conf. on pattern recognition.* IEEE.

Hanheide, M., Hofemann, N., & Sagerer, G. 2006. Action recognition in a wearable assistance system. *In: Proc. int. conf. on pattern recognition.* IEEE.

Hanheide, Marc. 2006 (dec). *A cognitive ego-vision system for interactive assistance.* phdthesis, Technische Fakultät – Universität Bielefeld.

Hanson, A., & Riseman, E. 1987. The VISIONS image understanding system. *In:* Brown, C. (ed), *Advances in computer vision.* Hillsdale, NJ: Erlbaum Press.

Heidemann, G., Bekel, H., Bax, I., & Ritter, H. 2005. Interactive online learning. *Pattern recognition and image analysis*, **15**, 55–58.

Henderson, J. M., & Hollingworth, A. 1999. High level scene perception. *Annual review of psychology*, **50**, 243–271.

Hitchcock, F. L. 1941. The distribution of a product from several sources to numerous localities. *Journal of math. phys.*, **20**, 224–230.

Hofemann, N., Fritsch, J., & Sagerer, G. 2004. Recognition of Deictic Gestures with Context. *Pages 334–341 of: Proceedings of the 26th dagm symposium on pattern recognition*, vol. 3175. Springer.

Hofemann, Nils. 2007. *Videobasierte handlungserkennung für die natürliche mensch-maschine-interaktion*. Dissertation, Universit
"at Bielefeld, Technische Fakult
"at.

Hoffmann, J., & Ziessler, M. 1986. The integration of visual and functional classifications in concept formation. *Psychol. res.*, **48**, 69–78.

Hofmann, Thomas, & Puzicha, Jan. 1998. *Statistical models for co-occurrence data*. Tech. rept. Massachusetts Institute of Technology, Cambridge, MA, USA.

Hoiem, D., Efros, A. A., & Hebert, M. 2005 (Oct.). Geometric context from a single image. *Pages 654–661 of: Proc. of int. conf. on computer vision (iccv)*, vol. 1.

Hoiem, Derek, Efros, Alexei A., & Hebert, Martial. 2006 (June). Putting objects in perspective. *Pages 2137–2144 of: Proc. ieee computer vision and pattern recognition (cvpr)*, vol. 2.

Hollingworth, Andrew, & Henderson, John M. 1998. Does consistent scene context facilitate object perception? *Journal of experimental psychology: General*, **127**(4), 398–415.

Isard, M., & Blake, A. 1998. Condensation – conditional density propagation for visual tracking. *International journal of computer vision*, **29**(1), 5–28.

Isard, Michael, & Blake, Andrew. 1996. Contour tracking by stochastic propagation of conditional density. *Pages 343–356 of: Europ. conf. on computer vision (eccv)*, vol. 1.

Itty, L., & Koch, C. 2001. Computational modelling of visual attention. *Nature reviews neuroscience*, **2**, 194–203.

Jackendoff, Ray. 1987. On beyond zebra: The relation of linguistic and visual information. *Cognition*, **26**, 89–114.

Jamieson, Michael, Dickinson, Sven, Stevenson, Suzanne, & Wachsmuth, Sven. 2006. Using language to drive the perceptual grouping of local image features. *Pages 2102–2109 of: Cvpr '06: Proceedings of the 2006 ieee computer society conference on computer vision and pattern recognition*. Washington, DC, USA: IEEE Computer Society.

Jamieson, Michael, Fazly, Afsaneh, Dickinson, Sven, Stevenson, Suzanne, & Wachsmuth, Sven. 2007. Learning structured appearance models from captioned images of cluttered scenes. *Pages 1–8 of: Computer vision, 2007. iccv 2007. ieee 11th international conference on*.

Jelinek, F. 1985. The development of an experimental discrete dictation recogniser. *Proceedings of the ieee*, **73**(11).

Jelinek, F., & Mercer, R. 1980. Interpolated estimation of markov source parameters from sparse data. *In: Proc. of the workshop of pattern recognition in practice*.

Jensen, F. V., & Nielsen, T. D. 2007. *Bayesian networks and decision graphs (2nd ed.)*. Springer.

Johnson-Laird, P. N. 1983. *Mental models: Towards a cognitive science of language, inference, and consciousness*. Cambridge, MA: Harvard University Press.

Just, M. A., & Carpenter, P. A. 1976. Eye fixations and cognitive processes. *Cognitive psychology*, **8**, 441–480.

Kaelbling, Leslie Pack, Littman, Michael L., & Cassandra, Anthony R. 1998. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, **101**(1-2), 99–134.

Kalman, R. E. 1960. A new approach to linear filtering and prediction problems. *Transactions of the asme-journal of basic engineering*, **82**(D), 35–45.

Käster, T., Pfeiffer, M., Bauckhage, C., & Sagerer, G. 2004. Intelligent navigation in image databases. *Ki zeitschrift*, **18**(4), 24–43.

Katz, S. M. 1987. Estimation of probabilities for sparse data for the language model component of a speech recogniser. *Assp*, **35**(3), 400–401.

Ke, Y., & Sukthankar, R. 2004. Pca-sift: A more distinctive representation for local image descriptors. *Pages 506–513 of: Proc. of ieee conf. on computer vision and pattern recognition*, vol. 2.

Keefe, D. E., & McDaniel, M. A. 1993. The time course and durability of predictive inferences. *Journal of memory and language*, **32**, 446–463.

Kintsch, W. 1998. *Comprehension: A paradigm for cognition*. Cambridge, UK: Cambridge University Press.

Kintsch, W., & van Dijk, T. A. 1978. Toward a model of text comprehension and production. *Psychological review*, **85**, 363–394.

Kittler, J., & Hancock, E.R. 1989. Combination evidence in probabilistic relaxation. *Int. journal of pattern recognition and artificial intelligence*, **3**(1), 29–51.

Kolonias, I., Yan, F., Christmas, W.J., Kostin, A., & Kittler, J. 2007a. A contextual reasoning framework for scene interpretation and tracking in tennis video sequences. *Computer vision and image understanding*. to appear.

Kolonias, I., Kittler, J., Christmas, W.J., & Yan, F. 2007b. Improving the accuracy of automatic tennis video annotation by high level grammar. *Pages 154–159 of: 14th int. conf. on image analysis and processing workshops*.

Kosecká, J., Zhou, L., Barber, P., & Duric, Z. 2003. Qualitative image based localization in indoor environments. *Pages 3–10 of: Ieee conf. on computer vision and pattern recognition*.

Kosslyn, S. M. 1994. *Image and brain.* Cambridge, MA: MIT Press.

Kummert, Franz. 1997. *Interpretation von bild- und sprachsignalen.* Shaker Verlag.

Laird, John E., Newell, Allen, & Rosenbloom, Paul S. 1987. Soar: An architecture for general intelligence. *Artificial intelligence*, **33**(1), 1–64.

Larman, Craig. 2005. *Applying uml and patterns : an introduction to object-oriented analysis and design and iterative development.* Upper Saddle River, NJ: Prentice Hall PTR.

Lazebnik, Svetlana, Schmid, Cordelia, & Ponce, Jean. 2006. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *Pages 2169–2178 of: Proc. of the ieee conf. on computer vision and pattern recognition*, vol. 2.

Li, S. Z. 1995. *Markov random field modeling in computer vision.* Computer Science Workbench, vol. XVI. Springer.

Li, Zhe, Fritsch, Jannik, Wachsmuth, Sven, & Sagerer, Gerhard. 2006. An object-oriented approach using a top-down and bottom-up process for manipulative action recognition. *Pages 212–221 of:* Franke, Katrin, Möller, Klaus R., Nickolay, Bertram, & Schäfer, Ralf (eds), *Pattern recognition, proc. of dagm*, vol. 4174. Heidelberg, Germany: Springer-Verlag.

Li, Zhe, Wachsmuth, Sven, Fritsch, Jannik, & Sagerer, Gerhard. 2007. View-adaptive manipulative action recognition for robot companions. *In: Int. conf. on intelligent robots and systems (iros).* San Diego, CA, USA: IEEE.

Lipson, Pamela R. 1996. *Context and configuration based scene classification.* Ph.D. thesis, MIT.

Lowe, David G. 1999 (September). Object recognition from local scale-invariant features. *Pages 1150–1157 of: Proc. international conference on computer vision.*

Lütkebohle, Ingo, Wrede, Sebastian, & Wachsmuth, Sven. 2005. Unsupervised filtering of xml streams for system integration. *Pages 204–212 of: Proc. of pattern recognition in information systems (pris 2005).*

Lux, A. 2003 (April). The imalab method for vision systems. *In: Int. conf. computer vision systems.*

Mackworth, Alan K. 1977. Consistency in networks of relations. *Artificial intelligence*, **57**(1), 99–118.

Marr, David. 1982. *Vision.* San Francisco: W. H. Freeman and Company.

Matas, J., Young, R., & Kittler, J.V. 1998. Hypothesis selection for scene interpretation using grammatical models of scene evolution. *Pages Vol II: 1718–1720 of: Proc. int. conf. pattern recognition (icpr-98).*

Matas, J., Chum, O., Urban, M., & Pajdla, T. 2002. Robust wide baseline stereo from maximally stable extremal regions. *Pages 384–393 of: British machine vision conf. (bmvc).*

McDermott, John P. 1982. R1: A rule-based configurer of computer systems. *Artificial intelligence*, **19**(1), 39–88.

McKeown, D. M., Harvey, W. A., & McDermott, J. 1985. Rule-based interpretation of aerial imagery. *Ieee trans. on pattern recognition and machine intelligence*, **7**(5), 570–585.

Melamed, Dan. 1997. Automatic discovery of non-compositional compounds in parallel data. *In: 2nd conference on empirical methods in natural language processing (emnlp).*

Mikolajczyk, K., & Schmid, C. 2002. An affine invariant interest point detector. *In: Proc. europ. conf. on computer vision (eccv).* Springer-Verlag.

Minsky, M. 1975. A framework for representing knowledge. *In:* Winston, P.H. (ed), *The psychology of computer vision.* McGraw-Hill, N.Y.

Minsky, Marvin. 1988. *The society of mind.* New York: Simon and Schuster.

Moore, D., Essa, I., & Hayes, M. 1999 (March). Exploiting human actions and object context for recognition tasks. *In: Proceedings of ieee international conference on computer vision.*

Moore, Darnell J. 2000. *Vision-based recognition of actions using context.* Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA.

Moringen, J., Wachsmuth, S., Dickinson, S., & Stevenson, S. 2008 (June). Learning visual compound models from parallel image-text datasets. *In: 30th annual symposium of the german association for pattern recognition (dagm).*

Moringen, Jan. 2007. *Lernen von wort-form-korrespondenzen aus bildern und bildunterschriften.* Tech. rept. Bielefeld University.

Murphy, P., Torralba, A., & Freeman, W. T. 2003. Using the forest to see the trees: a graphical model relating features, objects and scenes. *In: Adv. in neural information processing systems 16 (nips).* Vancouver, BC: MIT Press.

Nagel, H. H. 1988. From image sequences towards conceptual descriptions. *Image and vision computing*, **6**(2), 59–74.

Neuhaus, Michel, & Bunke, Horst. 2006. Edit distance-based kernel functions for structural pattern classification. *Pattern recogn.*, **39**(10), 1852–1863.

Newell, A., & Simon, H. 1972. *Human problem solving.* Englewood Cliffs, New Jersey: Prentice-Hall.

Nilsson, Nils. 1971. *Problem-solving methods in artificial intelligence.* New York, New York: McGraw-Hill.

Nistér, D., & Stewénius, H. 2006 (June). Scalable recognition with a vocabulary tree. *Pages 2161–2168 of: Ieee conference on computer vision and pattern recognition (cvpr)*, vol. 2.

Oatley, K. 1999. Why fiction may be twice as true as fact: Fiction as cognitive and emotional simulation. *Review of general psychology*, **3**, 101–117.

Ohta, Y. I. 1980. *A region-oriented image analysis system by computer.* Ph.D. thesis, Kyoto University.

Oliva, A. 2005. Gist of the scene. *Pages 251–256 of:* Itti, L., Rees, G., & Tsotsos, J. K. (eds), *The encyclopedia of neurobiology of attention.* San Diego, CA: Elsevier.

Oliva, A., & Torralba, A. 2001. Modeling the shape of the scene: a holistic representation of the spatial envelope. *Int. journal of computer vision (ijcv)*, **42**(3), 145–175.

Opelt, A., Fussenegger, M., Pinz, A., & Auer, P. 2004. Weak hypotheses and boosting for generic object detection and recognition. *Pages 71–84 of:* Pajdla, T., & Matas, J. (eds), *Eccv'04*. LNCS, vol. 3022. Springer.

O'Shaughnessy, D. 2000. *Speech communication : human and machine*. Addison-Wesley.

Palmer, S. E. 1975. The effects of contextual scenes on the identification of objects. *Memory and cognition*, **3**, 519–526.

Pantic, M., Pentland, A., Nijholt, A., & Huang, T.S. 2007. Human computing and machine understanding of human behavior: A survey. *Pages 47–71 of:* Huang, T.S., Nijholt, A., Pantic, M., & Pentland, A. (eds), *Artificial intelligence for human computing*. Lecture Notes in Artificial Intelligence, vol. 4451. Springer.

Pearl, J. 1988. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Palo Alto: Morgan Kaufmann Publishers.

Pickering, M., & Garrod, S. 2004. Toward a mechanistic psychology of dialogue. *Behavioral and brain sciences*, **27**, 169–226.

Pinhanez, C. S., & Bobick, A. F. 1998. Human action detection using pnf propagation of temporal constraints. *Pages 898–904 of: Computer vision and pattern recognition, 1998. proceedings. 1998 ieee computer society conference on.*

Ponweiser, W., Vincze, M., & Zillich, M. 2005. A software framework to integrate vision and reasoning components for cognitive vision systems. *Robotics and autonomous systems*, **52**(1), 101–114.

Potter, M. C., & Levy, E. I. 1969. Recognition memory for a rapid sequence of pictures. *Journal of experimenfal psychology*, **81**, 10–15.

Quillian, M. R. 1968. Semantic memory. *In:* Minsky, M. (ed), *Semantic information processing*. Cambridge: MIT Press.

Ribo, M., Brandner, M., & Pinz, A. 2003. A flexible software architecture for hybrid tracking. *Pages 1899–1906 of: Intervis*, vol. 3.

Rijsbergen, C. J. 1979. *Information retrieval*. London Boston: Butterworth.

Rimey, R. D., & Brown, C. M. 1992. Where to look next using a bayes net: Incorporating geometric relations. *Pages 542–550 of: European conference on computer vision*.

Rimey, R. D., & Brown, C. M. 1994. Control of selective perspection using bayes nets and decision theory. *Int. journal of computer vision*, **12**, 173–207.

Rosch, E., & Mervis, C. B. 1975. Family resemblances: Studies in the internal structure of categories. *Cognitive psychology*, **7**, 573–605.

Rosenfeld, A., Hummel, R., & Zucker, S. 1976. Scene labeling by relaxation operations. *Ieee transactions on systems, man and cybernetics*, 420–433.

Roy, Deb. 2003. Grounded spoken language acquisition: Experiments in word learning. *Ieee transactions on multimedia*, **5**(2), 197–209.

Roy, Deb, & Pentland, Alex. 2002. Learning words from sights and sounds: A computational model. *Cognitive science*, **26**(1), 113–146.

Rubner, Y., Tomasi, C., & Guibas, L. J. 2000. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, **40**(2), 99–121.

Russel, Stuart, & Norvig, Peter. 2003. *Artificial intelligence: A modern approach*. Prentice Hall.

Sagerer, G., & Niemann, H. 1997. *Semantic networks for understanding scenes*. Advances in Computer Vision and Machine Intelligence. New York: Plenum Publishing Corporation.

Salton, G. 1971. Experiments in automatic thesaurus construction for information retrieval. *Pages 43–49 of: Proc. ifip congress, ta-2*.

Schack, T., & Mechsner, F. 2006. Representation of motor skills in human long-term memory. *Neuroscience letters*, **391**, 77–81.

Schank, R. C., & Abelson, R. P. 1977. *Scripts, plans, goals and understanding.* Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Schneider, W., & Körkel, J. 1989. The knowledge base and text recall: Evidence from a short-term longitudinal study. *Contemporary educational psychology*, **14**, 382–393.

Schubert, Falk, Spexard, Thorsten, Hanheide, Marc, & Wachsmuth, Sven. 2007 (March). Active vision-based localization for robots in a home-tour scenario. *In: Int. conf. of computer vision systems (icvs).*

Schürmann, J. 1977. *Polynomklassifikatoren für die zeichenerkennung.* München: Oldenbourg Verlag.

Shapiro, L., & Haralick, R. 1985. A metric for comparing relational descriptions. *Ieee transactions on pattern analysis and machine intelligence*, 90–94.

Shi, Jianbo, & Malik, J. 2000. Normalized cuts and image segmentation. *Pattern analysis and machine intelligence, ieee transactions on*, **22**(8), 888–905.

Shoham, Yoav. 1999. What we talk about when we talk about software agents. *Ieee intelligent systems*, **14**(2), 28–31.

Shum, H.-Y., & Szeliski, R. 1997. *Panoramic image mosaics.* Tech. rept. Microsoft Research.

Siddiqi, K., Shokoufandeh, A., Dickinson, S., & Zucker, S. 1999. Shock graphs and shape matching. *International journal of computer vision*, **30**, 1–24.

Siegl, H., Schweighofer, G., & Pinz, A. 2004. An ar human computer interface for object localization in a cognitive vision framework. *Pages 176–186 of: Int. workshop on computer vision in human-computer interaction (eccv)*, vol. 3058. Springer.

Siegl, H., Hanheide, M., Wrede, S., & Pinz, A. 2007. An augmented reality human-computer interface for object localization in a cognitive vision system. *Image vision comput.*, **25**(12), 1895–1903.

Siskind, Jeffrey M. 2001. Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic. *Journal of artificial intelligence research*, **15**, 31–90.

Siskind, Jeffrey M., & Morris, Quaid. 1996. A maximum-likelihood approach to visual event classification. *Pages 347–360 of: ECCV (2)*.

Siskind, Jeffrey Mark. 1996. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition*, **61**(1-2), 1–38.

Sivic, J., & Zisserman, A. 2004. Video data mining using configurations of viewpoint invariant regions. *Pages I–488–I–495 Vol.1 of: Computer vision and pattern recognition, 2004. cvpr 2004. proceedings of the 2004 ieee computer society conference on*, vol. 1.

Sivic, Josef, & Zisserman, Andrew. 2003. Video google: A text retrieval approach to object matching in videos. *Pages 1470–1477 of: Proc. of int. conf. on computer vision (iccv)*.

Smith, J. R., & Chang, S. 1995 (Oct.). Single color extraction and image query. *Pages 528–531 of: Proc. of ieee int. conf. of image processing*, vol. 3.

Smith, J. R., & Chang, S. 1996 (Sep.). Local color and texture extraction and spatial query. *Pages 1011–1014 of: Proc. of ieee int. conf. on image processing*, vol. 3.

Socher, G., Sagerer, G., & Perona, P. 2000. Bayesian reasoning on qualitative descriptions from images and speech. *Image and vision computing*, **18**, 155–172.

Srihari, R.K. 1995. Computational models for integrating linguistic and visual information: A survey. *Artificial intelligence review*, **8**(5-6), 349–369.

Strat, T. M. 1993. Employing contextual information in computer vision. *Pages 217–229 of: Darpa93*.

Strat, Thomas M., & Fischler, Martin A. 1991. Context-based vision: Recognizing objects using information from both 2d and 3d imagery. *Ieee transactions on pattern analysis and machine intelligence*, **13**(10), 1050–1065.

Swadzba, Agnes, & Wachsmuth, Sven. 2008 (December). Categorizing perceptions of rooms using 3d features. *In: Proc. iapr int. workshops on statistical techniques in pattern recognition (spr 2008)*. submitted.

Szyperski, C. 1999. *Component software*. UK: Addison Wesley.

Therriault, David J., Rinck, Mike, & A., Zwaan Rolf. 2006. Assessing the influence of dimensional focus during situation model construction. *Memory & cognition*, **34**(1), 78–89.

Torralba, A., Murphy, K., Freeman, W., & Rubin, M. 2003. Context-based vision system for place and object recognition. *Pages 273–280 of: Proc. 9th ieee int. conf. on computer vision (iccv)*, vol. 1.

Torralba, Antonio. 2003. Contextual priming for object detection. *Int. j. comput. vision*, **53**(2), 169–191.

Tsotsos, J. K. 1989 (August). The complexity of perceptual search tasks. *Pages 1571–1577 of: Proc. international joint conference on artificial intelligence*.

Tsotsos, J. K. 1992. On the relative complexity of passive vs active visual search. *International journal of computer vision*, **7**(2), 127–141.

Tsotsos, John K., & Shubina, Ksenia. 2007 (March). Attention and visual search : Active robotic vision systems that search. *In: 5th international conference on computer vision systems conference*.

Turk, M. 2005. Multimodal human-computer interaction. *In:* Kisancanin, B., Pavlovic, V., & Huang, T. S. (eds), *Real-time vision for human-computer interaction*. Berlin, NY: Springer.

Turk, M., & Pentland, A. 1991. Eigenfaces for recognition. *Journal of cognitive neuro science*, **3**(1), 71–86.

Ungerleider, L. G., & Mishkin, M. 1982. Two cortical visual systems. *Pages 549–585 of:* Ingle, D. J., & Goodale, M. A. (eds), *Analysis of visual behavior*. Cambridge, Massachusetts: MIT Press.

van Dijk, T. A., & Kintsch, W. 1983. *Strategies in discourse comprehension*. New York: Academic Press.

Vasconcelos, Nuno. 2007. From pixels to semantic spaces: Advances in content-based image retrieval. *Computer*, **40**(7), 20–26.

Vincze, Markus, Ponweiser, Wolfgang, & Zillich, Michael. 2006. Contextual coordination in a cognitive vision system for symbolic activity interpretation. *In: Proc. of the fourth ieee int. conf. on computer vision systems (icvs)*.

Viola, Paul, & Jones, Michael. 2004. Robust real-time object detection. *International journal of computer vision*, **57**(2), 137–154.

Vorwerg, Constanze. 2001a. Kategorisierung von Grössen- und Formattributen. *In:* Zimmer, A., Lange, K., Bäuml, K., Loose, R., Scheuchenfpflug, R., Tucha, O., Schnell, H., & Findl, R. (eds), *Abstracts der 43. tagung experimentell arbeitender psychologen*. Experimentelle Psychologie. Lengerich: Pabst Science Publishers.

Vorwerg, Constanze. 2001b (Apr. 9–11). Kategorisierung von Grössen- und Formattributen. *In: Posterbeitrag auf der 43. tagung experimentell arbeitender psychologen*.

Wachsmuth, S., Wrede, S., Hanheide, M., & Bauckhage, C. 2005. An active memory model for cognitive computer vision systems. *Ki-journal, special issue on cognitive systems*, **19**(2), 25–31.

Wachsmuth, Sven. 2001. *Multi-modal scene understanding using probabilistic models*. Ibidem Verlag.

Wachsmuth, Sven, & Sagerer, Gerhard. 2002. Bayesian networks for speech and image integration. *Pages 300–306 of: Eighteenth national conference on artificial intelligence*. Menlo Park, CA, USA: American Association for Artificial Intelligence.

Wachsmuth, Sven, Stevenson, Suzanne, & Dickinson, Sven. 2003. Towards a framework for learning structured shape models from text-annotated images. *Pages 22–29 of: Proceedings of the hlt-naacl 2003 workshop on learning word meaning from non-linguistic data*. Morristown, NJ, USA: Association for Computational Linguistics.

Wachsmuth, Sven, Wrede, Sebastian, & Hanheide, Marc. 2007. Coordinating interactive vision behaviors for cognitive assistance. *Computer vision and image understanding*, **108**, 135–149.

Walker, E. L., Herman, M., & Kanade, T. 1988. A framework for representing and reasoning about three-dimensional objects for vision. *Ai magazine*, **9**(2), 47–58.

Westling, M., & Davis, L. 1996. Object recognition by fast hypothesis generation and reasoning about object interactions. *In: 13th international conference on pattern recognition (icpr)*.

Westling, M., & Davis, L. 1998. Interpretation of complex scenes using bayesian networks. *In: Asian conference on computer vision (accv'98)*.

Wiener, Norbert. 1948. *Cybernetics: Or the control and communication in the animal and the machine*. Cambridge, MA: MIT Press.

Winograd, Terry. 2006. Shifting viewpoints: Artificial intelligence and human-computer interaction. *Artificial intelligence*, **170**, 1256–1258.

Woods, W. A. 1975. What's in a link? foundations of semantik networks. *Pages 35–82 of:* Bobrow, B., & Collins, A. (eds), *Representation and understanding*. New York: Academic Press.

Wooldridge, M., & Jennings, N. 1995. Intelligent agents: theory and practice. *Knowl. eng. rev.*, **10**(2), 115–152.

Wrede, Sebastian, Hanheide, Marc, Wachsmuth, Sven, & Sagerer, Gerhard. 2007. Integration and coordination in a cognitive vision system. *In: Proc. of international conference on computer vision systems*. IEEE, St. Johns University, Manhattan, New York City, USA.

Wren, C. R., Azarbayejani, A., Darrell, T., & Pentland, A. 1997. Pfinder: Real-time tracking of the human body. *Ieee transactions on pattern analysis and machine intelligence*, **19**(7), 780–785.

Zhang, J., Marszalek, M., Lazebnik, S., & Schmid, C. 2005. *Local features and kernels for classification of texture and object categories: An in-depth study.* Tech. rept. INRIA.

Zwaan, R. A., Langston, M. C., & Graesser, A. C. 1995. The construction of situation models in narrative comprehension: An event-indexing model. *Psychological science*, **6**, 292–297.

Zwaan, R.A. 2004. The immersed experiencer: Toward an embodied theory of language comprehension. *The psychology of learning and motivation*, **44**.

Zwaan, Rolf A., & Radvansky, Gabriel A. 1998. Situation models in language comprehension and memory. *Psychological bulletin*, **123**(2), 162–185.

Index