A Hierarchical Markov Logic Based Framework for Reasoning with Incomplete Visual Evidence

A THESIS

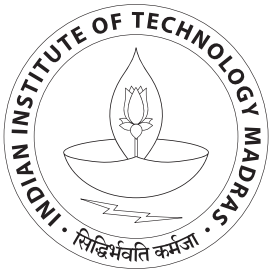
submitted by

PRIYA ANNA MANI

for the award of the degree

of

MASTER OF SCIENCE
(by Research)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, MADRAS.

June 2012

THESIS CERTIFICATE

This is to certify that the thesis entitled **A Hierarchical Markov Logic Based Framework for Reasoning with Incomplete Visual Evidence**, submitted by **Priya Anna Mani**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Science (by Research)**, is a bona fide record of the research work carried out by her under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. B. Ravindran
Research Guide
Associate Professor
Dept. of Computer Science and Engineering
IIT-Madras, 600 036

Place: Chennai

Date:

ACKNOWLEDGEMENTS

First and foremost, I thank Lord Almighty for showering His grace and wisdom upon me which helped me throughout my research. I express my sincere thanks to my advisor Dr. B. Ravindran, without whose support this work would not have been brought forth. His untiring support and guidance was present all throughout this work. He was always approachable, ready to clear my doubts and taught me a lot regarding research. I especially admire his patience in difficult times and express my heart-felt gratitude to him. I also thank him for giving me an opportunity to work with University of Birmingham, UK.

I thank Dr. Jeremy Wyatt and Jose Nunez-Varela of University of Birmingham for their discussions and collaboration with us during various stages of the research, especially while formulating the problem. I also thank the British Council for funding the UKIERI project which enabled me to do research with University of Birmingham. I thank Dr. Sriraam Natarajan of Wake Forest University for providing useful pointers on relational models and Markov logic networks and for clearing many of my doubts regarding them. I thank my GTC which include Dr. Kamala Krithivasan, Dr. Sutanu Chakraborti and Dr. V. Srinivasa Chakravarthy for their valuable suggestions regarding my work.

I thank RISE labmates whose support and acquaintance helped me focus on my work. I thank Yousuf for helping me in my first steps into research and for his knowledge transfer sessions with me. Pradyot and Manimaran helped in generating the image dataset using Microsoft's Kinect. Swapna, Shiva, Ranga and many others have been helpful in various ways while the research progressed. My hostelmates and friends in IITCF have been a constant support for me with their prayers and the friendship they shared with me. I thank all of them who have made my stay at IIT Madras a memorable one. I especially thank my parents for their constant support and help during my stay at IIT.

Finally, I thank the administration of IIT Madras and the CSE department for providing

me the facilities and environment for conducting research, especially the CSE department and Central libraries.

ABSTRACT

KEYWORDS: Probabilistic inference; Markov logic; Visual routines; Graphical models; Active vision; Object categorization.

Visual perception is a key function for an embodied agent to interact with its environment for complex object manipulation tasks. The theory of visual routines suggests a framework for employing perception to solve high-level vision tasks in a cognitively oriented way. But a major challenge in building vision systems for embodied agents is that the evidence obtained from sensors is uncertain and incomplete, i.e., the results of operation of visual routines are not completely reliable. This is due to the inherent limitations of the equipment in terms of field-of-view and resolution of the camera, which causes the input image to be of low fidelity. Moreover, the application of visual operators may yield spurious or imprecise evidence and choosing the right parameters is hard. We propose a novel approach for inference over uncertain and incomplete evidence, using Markov Logic Networks (MLN) and active vision in a hierarchical framework and evaluate it using an object categorization task.

Markov Logic Networks belong to the class of Statistical Relational Learning (SRL) methods that combine the expressiveness of first-order logic and the ability of probability theory to handle uncertainty. MLNs extend Markov networks to a relational setting by expressing the knowledge as a set of weighted formulas. We propose a layered MLN design which performs stage-wise inference to allow for reasoning at multiple levels and at varying levels of uncertainty. Given that the information is incomplete, active vision is a mechanism for focused gathering of additional information. Our framework integrates active vision with the layered MLN model to gather missing evidence, facilitating reliable and tractable inference. Inspired by the ideas of active vision, in the event of missing

evidence, our framework restricts the selective visual processing to specific regions of the input image and further inference is carried out incorporating the new evidence.

We present a cognitively motivated, complete end-to-end system for object categorization in a SRL framework. We use three different datasets for experimental evaluation: synthetic images generated using OpenCV library, images obtained from the iCub humanoid simulator and real images taken from Microsoft's Kinect Xbox (R). The system is evaluated with different levels of incompleteness and noise on these datasets and empirically prove its applicability to detect objects of complex structures.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

SRL	Statistical Relational Learning
MLN	Markov Logic Network
MRF	Markov Random Field
CRF	Conditional Random Field
PRISM	PRogramming In Statistical Modeling
BLP	Bayesian Logic Program
LBN	Logical Bayesian Network
RBN	Relational Bayesian Network
BP	Belief Propagation
MAP	Maximum A Posteriori
FOL	First Order Logic
OpenCV	Open Computer Vision
MC-SAT	Monte Carlo- SATisfiability
SCG	Scaled Conjugate Gradient
DN	Diagonal Newton
VP	Voted Perceptron
RGB	Red Green Blue
GLCM	Gray-Level Co-occurrence Matrix
RL	Reinforcement Learning

NOTATION

$\mathbf{E}_i$	Set of evidence nodes at layer *i*
$\mathbf{O}_i$	Set of output nodes at layer *i*
$\mathbf{e}_i^{LL}$	Evidence for layer *i* propagated from lower layers
$\mathbf{e}_i^{IL}$	Evidence for layer *i* generated in-layer
v_i	Visual routines associated with layer *i*
O_{ij}	Output node *j* at layer *i*
p_j	Probability of the label of output node O_{ij}
$evd(O_{ij})$	Subset of evidence required for inferring query *j* at layer *i*
$f_{i,i+1}$	Interface function between layer *i* and *i+1*
(O, p)	A particular output node and its probability
thresh	The probability threshold used to propagate evidence to higher layers

CHAPTER 1

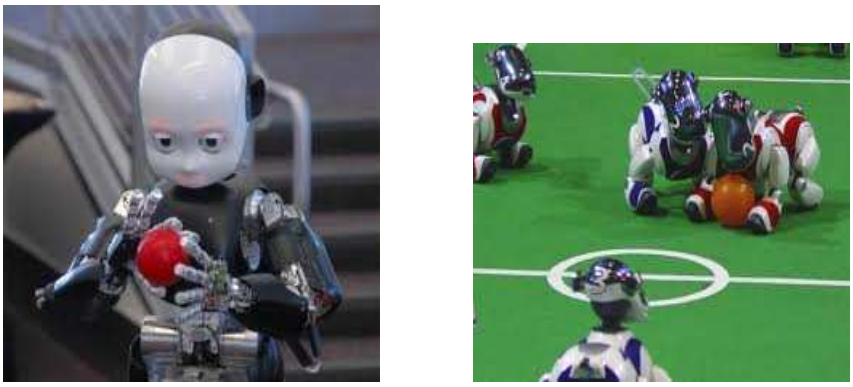
INTRODUCTION

1.1 Embodied Vision

Embodied vision systems are those which exist as one of the several functionalities of an embodied agent such as a robot or a human-being. Embodied agents have the property of situatedness, i.e., their perception and reachability of their environment is constrained by their physical location in the environment. But they have the ability to interact with their environment, which is often necessary and advantageous for the execution of their tasks. Some example tasks include robot navigation, obstacle avoidance, grasping objects, game playing, etc. In all these tasks the agent needs to continuously perceive its world in order to act upon it. It can also utilize its current perception of the world to actively control the next action to be taken or scene location to be perceived.

Vision based sensing and control is one of the several ways by which an agent can perceive its environment, the other methods being sonar, infra-red sensors, etc. Visual perception is the ability to identify and reason about objects in an agent's environment. Perception is one of the key requirements for cognitive behaviour. The tasks in which visual perception plays a major role in their execution are known as visually guided tasks. Few examples of visually guided tasks are given in Figure 1.1. Unlike traditional vision systems, embodied vision systems can provide vital information which can aid in the decision making of the agent and can also actively control the agent's actions.

Ullman (1984) proposed the theory of visual routines which states that visual perception requires the extraction of shape properties and spatial relations among objects and object parts. Visual routines are a set of efficient operations assembled from a fixed set of basic visual operators that can be used to extract high level concepts from the image.



(a) (b)

Figure 1.1: Examples of visually guided tasks: (a) an iCub robot trying to grasp an object, (b) a RoboCup soccer match.

These visuo-spatial relations are useful for handling a variety of complex object manipulation tasks, which can be performed efficiently by building the routines in a modular way. The visual routines theory is cognitively motivated and presents a model of how the vision system of human beings (or embodied agents) could work to solve high level vision tasks.

Research on visual cognition (Singh and Hoffman, 2001) provides empirical evidence that human vision system organizes shapes in terms of parts and their spatial relations. Objects have part structures which form perceptual units. These parts are computed automatically, quickly, and in parallel over the visual field. These part-based representations also explain the allocation of visual attention to objects.

1.2 Challenges

A major challenge in building vision systems for embodied agents is that the evidence obtained from the sensors is uncertain and incomplete. This is due to the inherent limitations of the equipment in terms of field-of-view and resolution of the camera, which causes the input image to be of low fidelity. The application of visual operators on the image will not yield all of the supportive evidence needed to determine an object category. The evidence

may contain the typical imprecision associated with vision processing. For example, a detected line may not lie between the exact corners of an object. Moreover, they may give spurious evidence and choosing the right parameters for the visual operators is hard. Other factors which influence vision processing are lighting conditions, perspective of the camera, etc. Thus, the evidence obtained is incomplete, inexact and inaccurate. Incomplete evidence may also result from occlusion of objects or partial visibility of the environment due to embodiment of the agent.

Hence, the result of the visual routines' operation on an image is unreliable, and object inference has to be performed over incomplete and uncertain evidence.

1.3 Motivation and Objective

Motivated by visual routines theory, we present a model for inference over uncertain and incomplete information generated by visual routines and evaluate it using an object categorization task for embodied agents. Object categorization plays an important role (either explicit or implicit) in all visually guided tasks involving embodied agents. The tasks require the agent to identify and act upon objects through continued interaction with its environment. In Figure 1.1, (a) represents explicit object detection while in (b) object detection is implicitly present in the various stages of the game, eg., tracking the ball, goal-post, opponent team, etc. We chose the object categorization task as it is an important step in all visual guided tasks and for cognitive behaviour. It involves perceptual grouping of visual information which can form an integral part of the decision making process of the agent. In recent years, there has been a lot of interest in developing *Statistical Relational Learning* (SRL) (Getoor and Taskar, 2007) methods that combine the expressiveness of first-order logic and the ability of probability theory to handle uncertainty. They extend traditional graphical models to a first-order representation, thus providing the ability to handle general relations between objects using a single template. The advantage of these models is that they can succinctly represent probabilistic dependencies between the attributes of different related objects, leading to sample-efficient learning and inference. These models allow for

reasoning at multiple-levels under varying levels of uncertainty. Our proposed work is an attempt at object categorization with incomplete evidence using basic visual features in a SRL framework.

We propose a hierarchical framework for object categorization from uncertain and incomplete evidence using SRL and active vision. Since the visual operators do not yield complete evidence, the inference has to be performed over uncertain evidence. Moreover, the object parts are inter-related and the spatial relationship among parts is a key feature for object detection. The advantage of SRL models is the ability to exploit such relationships while reasoning under uncertainty.

Specifically, we use Markov Logic Networks (Domingos and Lowd, 2009) (MLNs) for our design and implementation of the proposed system. MLNs extend traditional Markov networks to a relational setting by representing the features as a set of weighted rules in first-order logic. One of the nice features of MLNs is that they allow the user to write as many rules as possible about the domain and then learn weights for the rules to perform inference. This allows us to define the features associated with shapes and relations and form a hierarchical MLN that can reason at multiple levels (i.e., first reason about lines and circles, use the reasoning from that level to reason about l-shaped structures, then reason about squares and so on). More details on SRL models and their application in our work are discussed further in the remaining chapters of this thesis.

For reliable detection of objects, we use active vision to gather missing evidence whereby the visual processing is selectively applied to object parts from which more information is required to conclude their category. Active vision (Swain and Stricker, 1993) refers to mechanisms by which new information can be acquired autonomously through interaction with an agent's environment. An active vision system consists of two major components: (1) *visual behaviour*, which is a combination of primitive visual routines for the execution of a task (eg., pick-up object/put-down object) (2) *visual routines*, which forms a vocabulary of basic functions needed to build a vision system. The active vision paradigm provides the ability to combine vision with behaviour, which is vital to achieving robust execution of the agent's tasks. It is more robust than traditional vision techniques be-

cause the agent can improve its initial guess of the object category by obtaining additional evidence. It also helps in filtering irrelevant evidence through selective visual processing. Active vision techniques were chosen because they are robust and conform to the cognitive theory of vision. The inherent interactions of embodied agents makes it easily deployable on them.

The objectives of our work include the following:

- To develop a probabilistic relational framework for object detection
- To build a system which can take multiple observations of an image to actively control the inference of objects
- To develop a framework for hybrid bottom-up and top-down reasoning

Thus the scope of this work will be restricted to cognitively motivated architectures and not to compete with state of the art object detection algorithms.

1.4 Contributions of the Thesis

This thesis makes the following key contributions:

- A cognitively motivated, complete end-to-end system that takes an image as input and outputs the category and location of objects in the image. This system includes a pre-processing step that extracts the basic visual features from the image, a hierarchical MLN inference engine which outputs a distribution over the shapes inferred at each stage and an active vision component.
- The design of a hierarchical MLN model which performs stage-wise inference using evidence from lower levels to reason at higher levels. The MLNs handle the uncertainty in perception and inference of objects. To the best of our knowledge, this is the first work which explores MLNs in a layered architecture for any application.
- A method to integrate active vision with MLN inference facilitating reliable and tractable inference of the object category from incomplete evidence.
- Finally, the system is evaluated with different levels of incompleteness and noise on multiple datasets and the ability to identify complex shapes is established empirically.

1.5 Organization

The purpose of this chapter was to present an overview of embodied vision, the concept of visual routines and active vision and the need for reasoning with uncertain evidence. The objective of our work is also presented. The following chapters elaborate on our work and are organized as given below:

Chapter 2 gives a brief overview about the three major concepts employed in our work: visual routines, statistical relational models and active vision. The motivation for employing these methods in our work is also discussed. This is followed by an outline of the related work present in the literature.

Chapter 3 discusses the proposed system and its implementation. A complete end-to-end system for reasoning with incomplete and uncertain visual evidence is presented. The evolution of the idea is presented followed by an explanation of the general architecture of the system and its component modules. The chapter also describes the integration of active vision with the hierarchical MLN model for the task of object categorization.

Chapter 4 presents the experimental evaluation of the proposed system for categorization of objects with geometric regularities. The system is evaluated on on three different datasets: synthetic images generated using OpenCV, images obtained from the iCub humanoid simulator and real images taken from Microsoft's Kinect. The performance of the system for different levels of incomplete and noisy evidence is presented. The evaluation of the system on embodied agents is presented through the experiments on iCub and Kinect images. Comparison of the system with a baseline detector is also presented.

Chapter 5 summarizes the work carried out and the conclusions drawn from the thesis as a whole, followed by an outline of future research directions.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter, we give an overview of the three major concepts involved in our work: the theory of visual routines, relational models and active vision paradigm. We explain these concepts from the perspective of object categorization and give an intuition on why and how they can be useful for an embodied vision system. Related work on these topics are also presented towards the end of this chapter.

2.1 Visual Routines Theory

Most vision related tasks require the extraction of *visuo-spatial* relations, i.e., they require the identification of entities (which may be persons, static or dynamic objects, etc) in the scene and the relationships between them. Few examples of tasks which require visuo-spatial analysis are given in Figure 2.1. For example, in Figure 2.1, in order to answer the query (a) *How many cups are there in the scene?*, one has to identify the entities *cup* and *saucer* and the relation *on top of* between them. The same reasoning applies to the other tasks also. In (b) the fruit basket is expected to be on a table-top or a horizontal surface as opposed to the ceiling, while in (c) the road-signs have to be identified and the relation between their directions have to be established.

The above tasks seem effortless for humans but it hides a complex array of processes. (Ullman, 1984) proposed the visual routines theory as a way to compute visuo-spatial relations efficiently and to explain intermediate vision in human-beings. He suggests that visual perception requires the ability to extract shape properties and spatial relations. The visual routines are defined as efficient sequences of basic visual operations which can establish the visuo-spatial relations and can be used to build complex vision systems in a modularized way. Ullman suggests the following stages for visual perception:

(a) (b) (c)

Figure 2.1: Examples of tasks that use visuo-spatial analysis: (a) How many cups are there in the scene? (b) Where is the fruit basket? (c) Is the Opera House in the same direction as the Youth Hostel?

- Development of base representations of the environment in a bottom-up, spatially uniform way.
- Application of visual routines on the base representations to form incremental representations.
- Application of costly or task-specific routines on the incremental representations.

Different routines may share the same basic operations, thus building the system in a modularized way. The same routine may be applied to different spatial locations in parallel. The initial vision processing is applied across the visual field to form the base representation. It identifies regions of interest (salient regions) for focused visual processing using visual routines, which forms subsequent incremental representations. An example of perception through visual routines is shown in Figure 2.2. Inspired by this concept, we define objects as being composed of sub-parts. For example, two *lines* intersect to form an *L-structure*, two *L's* intersect to form a *square*, a *square-face* and an *L-face* forms a *cube*, etc. At each layer of the proposed hierarchical model, we define visual routines which are applied on object parts detected from lower layers to identify the shape and spatial properties associated with that layer.

For applying visual routines, mechanisms are required for selecting the locations at which they should be applied and sequencing of the operations to extract relevant infor-

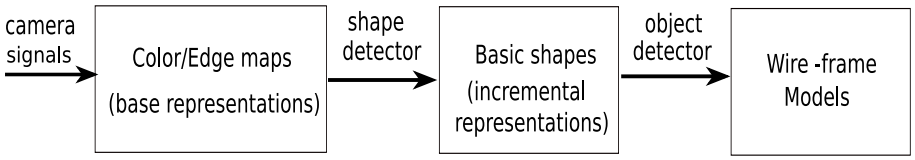


Figure 2.2: Example of visual perception with visual routines: the edge and color detectors are applied across the entire image to identify salient regions. Higher-level visual routines are applied only on these regions and are build systematically to form wire-frame object models.

mation at minimum cost. The basic operations need to be identified and integrated into meaningful visual routines and a control mechanism is needed to decide when and where visual routines are to be applied.

2.1.1 Evidence from Cognitive Research

This sub-section discusses some of the research work in neuroscience which is supportive of the visual routines theory. Vision processing has long been considered to be a bottom-up process. (Marr, 1976) states that early vision processing generates a rich description of primitive gray-level changes in an image, represented as a primal sketch. The description is expressed as *edge, line, blob, etc.* Grouping operations on the primal sketch determines higher level objects. (Biederman, 1987) suggests that visual information obtained from an image is divided into simple geometric components known as *geons* which is then matched with the most similar object representation known. Recent research in neuroscience support the theory that human representation of visual shape is part-based. (Singh and Hoffman, 2001) suggests that perceptual units occur not only at the object level but also at the part level. The objects are represented using parts and their relationships. Empirical evidence from (Baylis and Driver, 1994) suggests that parts are computed automatically, quickly, and in parallel over the visual field. (Barenholtz and Feldman, 2001) and (Singh and Scholl, 2000) provide evidence for attentional shifts within single objects (part based attention). There is also empirical evidence for activation of human cortical regions by stereoscopically defined object shapes (Gilaie-Dotan *et al.*, 2001).

2.2 Methods for Uncertain Reasoning

In the previous section, we discussed the concept of visual routines and how they can be employed in perception. But, as discussed in Chapter 1, the result of operation of visual routines on an image is not completely reliable. Hence we need to reason with uncertain evidence to determine the objects present in an image. In this section, we present an overview of the methods for uncertain reasoning:

- **Probability:** The basic statistical method for handling uncertainty is by the axioms of probability. The axioms help in restricting the set of beliefs that an agent can hold in a domain. The Bayes' theorem provides a formal way to find the conditional probability of a hypothesis being true given the evidence. The rule is stated as follows for multi-valued variables:

$$\mathbf{P}(Y|X) = \frac{\mathbf{P}(X|Y)\mathbf{P}(Y)}{\mathbf{P}(X)}. \tag{2.1}$$

  The Naive Bayes' classifier is based on the Bayes' rule and provides a simplification of the computation using the class-conditional independence. For example, let the class variable to be determined be *C* which can take values in the range 0, 1, ..., *j* and the evidence variable *X* be a *d*-dimensional vector ($x_1, x_2, ..., x_d$). The classifier assigns a data point to class C_j such that

$$C_j = \max_j p(C_j|x_1, x_2, ..., x_d) = \max_j p(x_1, x_2, ..., x_d|C_j)p(C_j). \tag{2.2}$$

  The Naive Bayes' classifier assumes that the dimensions of the input (evidence) are independent of each other given the value of the class variable. Hence the above equation can be simplified using the equation:

$$p(x_1, x_2, ..., x_d|C_j) = \prod_{k=1}^{d} p(x_k|C_j). \tag{2.3}$$

  Thus probability theory provides a simple and principled mechanism to find unknown probabilities of variables (hypothesis) given the values for other variables (evidence).

- **Belief Networks:** The Naive Bayes' assumption does not hold in many real domains. For example, in image processing, the value of a pixel is best determined by taking into account the values of the neighbouring pixels also because pixels with spatial proximity are likely to have similar values. Belief networks (also known as Bayesian networks) model the causal influence between variables and removes the

class-conditional independence assumption. They fall under the general category of *graphical models* which represent the dependencies between the input dimensions as a graph and define factorizations of the joint probability distribution of variables by exploiting the local structures within the graph. The graph may be directed or undirected depending on whether the relationship between variables is a causal relationship or not. Thus they provide a more powerful and accurate inference mechanism than Naive Bayes' and are suitable for modeling real-world domains. A detailed explanation of graphical models and inference mechanisms is given in the following section.

- **Dempster-Shafer Models:** Dempster-Shafer models define a belief function to compute the probability that the evidence supports a proposition than the probability of the proposition itself. They address the problem of ignorance as opposed to uncertainty. These models are based on obtaining degrees of beliefs for a query variable based on subjective probabilities for a related variable. The Dempster's rule provides a means to combine such degrees of beliefs when they are based on independent evidence.

- **Fuzzy Sets:** Fuzzy set theory is a means of specifying the degree of vagueness of a variable. For example, *Tall* can be considered as a fuzzy predicate with *Tall (Bob)* having a value between 0 and 1 to indicate the degree of *tallness* than just *true* or *false*. Fuzzy logic is a method for reasoning with logical expressions describing membership in fuzzy sets.

In our work, we use graphical models for uncertain reasoning since they can model the dependencies between variables. The label of a node is influenced by the labels and attributes of its surrounding nodes. This is a key requirement for most real-world tasks and especially in image processing. The following section explains graphical models in general and the specific model used in our work.

2.3 Graphical Models

As explained in Section 1.2, the evidence extracted by the visual routines is not completely reliable. This is due to the inherent limitations of the sensor equipment and the image processing algorithms. Also, the task of object categorization can be naturally decomposed into categorization of sub-parts and establishing the spatial relations between them. Moreover, the performance of object categorization can be improved significantly by consider-

ing the relationship between other object parts as a feature than using the features of the object alone. Thus we need to:

- Reason from incomplete evidence
- Exploit relationship between object parts, i.e., model non-i.i.d. data

Various methods for reasoning with incomplete evidence were presented in Section 2.2. Among them, probabilistic graphical models stand out as a suitable method that satisfies both of the above requirements. They can reason about uncertain evidence in a principled way using probability theory and they inherently model the relationships between the variables in a domain. Graphical models use a graph-based representation as the basis for encoding complex distributions and to exploit the structure and interactions between variables in a domain. Graphical models may be directed or undirected. Directed models represent causal relationships between variables using a directed graph representation. Undirected models are suitable to represent interactions that are not necessarily causal in nature. Some applications which need undirected graphical models are vision related tasks such as image de-noising, segmentation, text processing tasks, etc. For example, in image de-noising and segmentation, the value of an image pixel is likely to be similar to the values of its neighbouring pixels though a pixel does not cause another pixel to be of a particular value. Hence the relationship between pixels have to be modeled as an undirected graph. The graph-based structure allows for efficient inference of the probabilities of the variables in the domain. In the next section we describe Markov Random Fields (MRF), an undirected graphical model as an example. We chose to give brief description of MRF (or Markov networks) since the relational model used in our work is an extension of this model.

2.3.1 Markov Random Fields

Markov Random Fields models the joint distribution of a set of variables X = (X_1, X_2, ..., X_n). The nodes in the network represent variables in the domain. An example MRF with

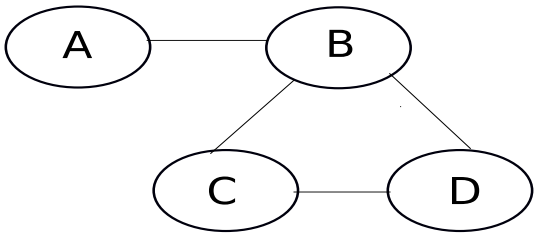


Figure 2.3: An example Markov Random Field

four variables in the domain is shown in Figure 2.3. The probabilistic interaction between the variables are captured by *potential functions* (Φ) defined over cliques in the graph.

According to *Hammersley-Clifford theorem*, if the distribution is strictly positive, then it can be factorized as a product of potential functions defined over cliques in the graph. The joint distribution can then be represented as:

$$P(X) = \frac{1}{Z} \prod_{c} \Phi_c(X). \tag{2.4}$$

where *Z* is a normalizing constant. The above equation can be represented in log-linear form as:

$$P(x) = \frac{1}{Z} exp \left(\sum_{j} w_j f_j(x) \right). \tag{2.5}$$

where *j* iterates over the cliques in the graph and f_j is the feature defined over clique *j* comprising of the subset of variables *x* from the set *X*.

The MRF also encodes conditional independences between variables for efficient inference. Two variables are independent of each other if the nodes along the path between them in the graph are marked as evidence. For example, in Figure 2.3, *A* is independent of *C* or *D* given *B*, i.e., the information about *A* does not add any more information about *C* or *D*, if state of *B* is known for sure. The set of nodes whose value when known makes a node independent of other nodes in the network, is called the *Markov blanket* of a node. For an MRF, the Markov blanket of a node is the set of its immediate neighbours. Inference

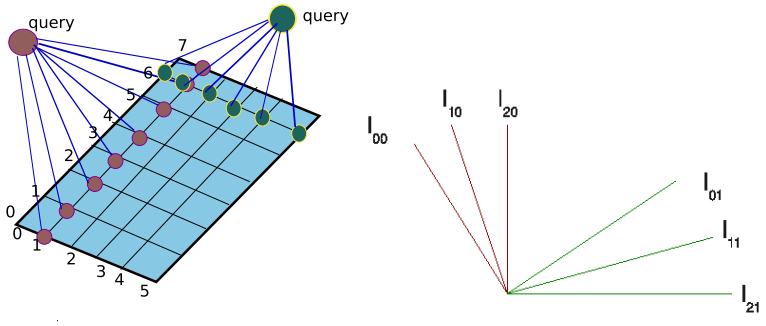
over MRFs can be done using a variety of algorithms such as variable elimination, belief propagation, sampling methods, etc. We will be discussing belief propagation algorithm in a further section.

2.3.2 Why Relational Model?

Many variants of graphical models have been proposed in literature such as (Conditional Random Fields) CRFs, discriminative CRFs, etc. But there is a major drawback with these traditional models: they have a rigid structure and cannot represent variable number of objects and general relations between objects in a domain. As an example, consider the task of detecting horizontal and vertical lines across an image as shown in Figure 2.4 (a). One possible design of a traditional model such as MRF is shown in the figure, where the image pixels are the evidence nodes which indicate whether the pixel is part of line or not. The evidence are connected to a query node and potential functions are defined over the binary cliques. But in order to detect lines across the image, this model has to be replicated across the image. This makes it tedious and unattractive to model higher-level structures. Instead, a relational model can detect objects across the image in parallel using a single template.

As another example, consider using a non-relational model such as a CRF to find perpendicular lines that intersect to form `L´s. One CRF would be needed to infer each possible orientation of the lines. Note that the first line could be at any angle of rotation from horizontal axis and the second line should be perpendicular to the first, as shown in Figure 2.4 (b). Also note that the lines need not be exactly perpendicular for visual perception. If we are to represent all possible orientations of the lines the number of parameters in the CRF would become prohibitively high rendering inference intractable. On the other hand, relational models allow the use of a single template to capture all possible orientations of the lines due to their ability to succinctly capture generalizations of the rotation angles.

We use Markov Logic Networks (Domingos and Lowd, 2009) (MLNs) as the relational model for our design and implementation of the proposed system. MLNs extend Markov



(a) (b)

Figure 2.4: (a) Detection of lines across an image using a non-relational model (b) Different orientations of perpendicular lines intersecting to form L's. Note that the lines need not be exactly perpendicular for visual perception (eg., I_{00}, I_{01}).

networks to relational setting by expressing the knowledge as a set of weighted formulas. One of the nice features of MLNs is that they allow the user to write as many rules as possible about the domain and then learn weights for the rules to perform inference. This allows us to define the features associated with shapes and relations and form a hierarchical MLN that can reason at multiple levels (i.e., first reason about lines and circles, use the reasoning from that level to reason about l-shaped structures, then reason about squares and so on). While we use MLNs in this work, the concepts can be extended to most SRL systems such as PRISM (Sato and Kameya, 2001), Problog (Raedt *et al.*, 2007), BLPs (Kersting and Raedt, 2007), LBNs (Fierens *et al.*, 2005), RBNs (Jaeger, 2007), etc. These systems are mostly equivalent (Jaeger, 2008; Bruynooghe *et al.*, 2009) for the application that we are considering in this work.

Although we can use MLNs or any SRL models for our task, the issue is the size of such an MLN (or any SRL model) and the complexity in inference. If a single monolithic MLN is used to infer over the entire image (i.e., identify lines, circles, their intersections and the more complex objects arising from their interactions), inference can become computationally intractable easily. Also, learning of such a model requires exponentially many examples. One of the important features of object recognition is that we can divide the problem into tasks at different levels. i.e., we can perform inference on parts of an object

at lower levels and then use the results of the inference at lower level parts to reason about higher level parts. For instance, it is quite natural that we can infer about lines at lower levels and use the result to infer about L-shapes and then to rectangles. In this work we propose to exactly use this intuitive idea for performing inference in SRL models particularly using MLNs. We give a brief overview of MLNs and the belief propagation algorithm in the next section.

2.3.3 Markov Logic Networks

One of the most popular and general SRL representations is *Markov Logic Networks (MLNs)* (Domingos and Lowd, 2009). MLNs provide an efficient way of combining probability and logic to handle uncertain and complex environments. An MLN consists of a set of formulas in first-order logic and their real-valued weights, $\{(w_i, f_i)\}$. Each formula, represented by a set of predicates and their connectives, specifies a constraint that should hold over the evidence in a domain. The weight for the formula specifies how hard the constraint is. From the perspective of object detection, each formula is a specification of an object or an object part as being composed of its sub-parts through certain relational operators. The constants are the set of objects and object parts in the domain.

An MLN can be viewed as a template for constructing Markov networks. We can instantiate an MLN as a Markov network with a node for each ground predicate (atom) and a feature for each ground formula. The network generated by assigning constants to the predicate variables of MLN rules is called a ground network. The set of ground predicates which occur together in an MLN formula form a clique in the ground network. An example of a clause and its grounding from the point of view of object detection is shown in Figure 2.5.

All groundings of the same formula are assigned the same weight, leading to the following joint probability distribution over all atoms:

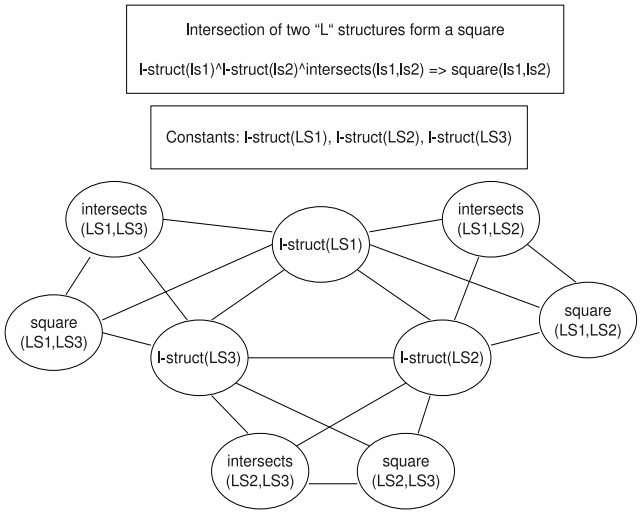


Figure 2.5: An example Markov logic rule and its ground network for a domain with three constants.

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right). \tag{2.6}$$

where $n_i(x)$ is the number of times the ith formula is satisfied by possible world x and Z is a normalization constant (as in Markov networks). Intuitively, a possible world where formula f_i is true one more time than a different possible world is e^{w_i} times as probable, all other things being equal. For this thesis, we will assume a finite set of atoms, but MLNs can also be defined over some infinite domains (Singla and Domingos, 2007).

In our framework, the object models composed of shape properties and spatial relations, are represented using first-order logic rules. The rules essentially represent the composition of visual routines needed to establish these properties in order to infer an object or its sub-part. The inference of the object is split across multiple layers, with an MLN at each layer to infer parts of objects. The object parts inferred at lower layers are combined to form the whole object at a higher layer. Layering makes our system tractable, since inference at higher layers occur only if supportive evidence is obtained from lower layers.

Note that the proposed hierarchical MLN is not equivalent to a single large MLN. The

MLNs at each layer performs independent inference in a sequential manner. We introduce new predicates from the conclusion of one level to the next. For example, a square is composed of two intersecting L's and L's are composed of two lines. The main advantage of this design is that it separates the inference of lower level entities such as lines and circles from higher-level complex shapes such as a train or a cube. The other advantage is that this modular design makes it possible to perform tractable inference since the individual layers are themselves significantly smaller than the original MLN.

Inference can be performed as a MAP estimate or by computing marginal probabilities. In the next section, we describe the belief propagation algorithm over an MLN.

2.3.4 Belief Propagation

Belief propagation is an efficient algorithm for computing marginal probabilities of nodes, i.e., the conditional probability of the query node is found given the values of the evidence nodes, by summing out over the other variables. The graph is first converted into a factor graph over which the algorithm is applied. A factor graph is a bipartite graph that expresses the structure of the factorization of the joint distribution given in equation 2.2. It consists of one variable node for each variable and a factor node for each local function over a subset of variables. A factor g_j is connected to a variable node x_i if and only if x_i is an argument of the local function corresponding to the factor g_j. An example factor graph of the ground network in Figure 2.5 is shown in Figure 2.6.

The belief propagation algorithm iteratively calculates the marginal probability of a node by passing *messages* between factors and variables. Equations 2.7 and 2.8 describe the messages passed for a non-relational graphical model. The message from a node to a factor is given by:

$$\mu_{x \to g}(x) = \prod_{h \in N(x) \setminus \{g\}} \mu_{h \to x}(x). \tag{2.7}$$

and the message from a factor to a node is:

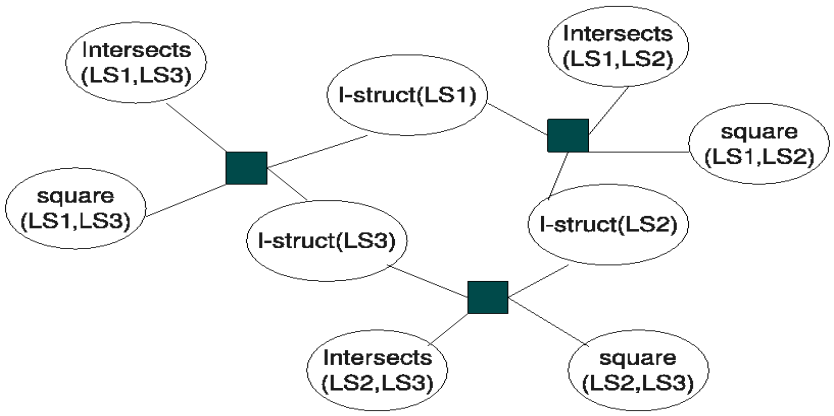


Figure 2.6: The factor graph of the ground network in Figure 2.5

$$\mu_{g\to x}(x) = \sum_{\sim\{x\}} \left(g(x) \prod_{y\in N(g)\setminus\{x\}} \mu_{y\to g}(y) \right). \tag{2.8}$$

where N(x) or N(g) is the set of neighbours of the variable or factor respectively. The algorithm is iterated until convergence, i.e., until the messages passed between a factor and a variable does not change. Convergence is guaranteed only for acyclic graphs, whereas for graphs with loops, *loopy* belief propagation can give approximate results.

Belief propagation in MLNs: Belief propagation in MLNs proceeds in the same way as described above, except that it is performed on a *minimal* graph. The minimal graph represents the smallest network of nodes required to answer a given query. The minimal graph is obtained as follows: add the query node into the network. Subsequently add its Markov blanket into the network and repeat this process until the node added is an evidence node. For example, the minimal network of Figure 2.5 needed to infer the query *P (square(LS1,LS2)| l-struct (LS1), l-struct (LS2))* is shown in Figure 2.7. This minimal network can be converted to a factor graph and belief propagation carried out on it.

2.3.5 Weight Learning in MLN

In this section we give a brief overview of the MLN weight learning methods employed in our work. As described earlier, MLNs can be described as a set of weighted formulae. The

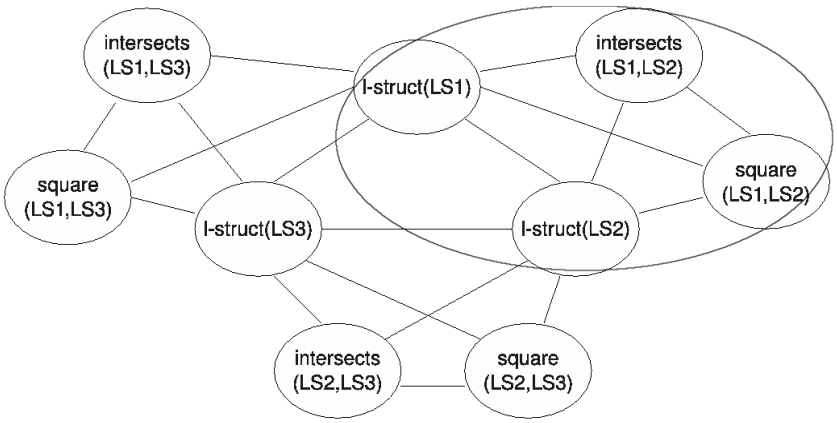


Figure 2.7: The minimal network (circled) to infer the query *P (square (LS1,LS2)| l-struct (LS1), l-struct (LS2))*

weight for a rule defines how hard the constraint is. Weight learning attempts to find the maximum *a posteriori* weights, i.e., the weights that maximize the product of their prior and likelihood from the data. But computation of the partition function of the likelihood is generally intractable. Hence gradient descent techniques are employed for learning the weights. The weight vector is updated at each step according to the formula:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{g} \tag{2.9}$$

where η is the learning rate and $\mathbf{g}$ is the gradient.

We compare three weight learning methods in our work: Voted Perceptron, Diagonal Newton and Scaled Conjugate Gradient. These methods compute the derivative of the negative conditional log-likelihood (CLL) with respect to a weight. For MLNs, this turns out to be the difference of the expected count of true groundings of a clause and the actual count. Voted perceptron approximates the expectation as the counts in the most probable explanation (MPE) state. In diagonal Newton method, the learning rate is replaced by the diagonalized Hessian of the negative CLL for faster convergence to a global minimum. Scaled conjugate gradient further speeds up the gradient descent by imposing a constraint that at each step, the gradient along the previous directions remain zero. Thus the effect of previous steps are not undone by the current step. More details on these methods can be

found at (Lowd and Domingos, 2007).

2.4 Active Vision

The previous sections discussed models for probabilistic inference from visual evidence in a systematic way. But a mechanism is needed for reliable inference in the presence of uncertain and incomplete evidence. Moreover, processing of the entire visual field is not efficient. Hence there should be a mechanism which can focus the visual processing to relevant parts of the image. Active vision techniques resolve these problems efficiently.

Active vision (Swain and Stricker, 1993) provides mechanisms for an agent to perceive its environment through continued interaction with it. It involves tight coupling of the agent's actions and perceptions of the world. The agent receives visual feedback which may be used to direct its next action. The new information may be obtained by changing camera parameters like focus, vergence, orientation, or it may involve selective sensing: in space, resolution and time. For example, new information may be obtained by zooming in on a part of the scene or by getting a different perspective of an object by rotating the object, shifting the view-point, etc. They may also involve features like spatially variant sensors (foveal sensors). Thus they help in removing the ambiguity associated with single shot images, which makes it more robust than traditional computer vision methods. Another characteristic of active vision systems is that they are naturally suited for embodied agents. Selective attention, an active vision technique, greatly simplifies computational costs by allowing processing at high resolution at desired regions of the image. In the active-vision paradigm, the main components of the vision system are termed *visual behaviours* and *visual routines*. Visual behaviours are combinations of primitive visual routines that help in executing a task (eg., pick-up object/put-down object) while visual routines form a vocabulary of basic operators for building the vision system. An interesting research area related to this is eye-hand co-ordination, where the visual system directs the hand (motor system) to specific locations in the world. For example, the hand may rotate an object to obtain a 3-D view and to increase the certainty of the object's identity

and affordance. They may also be implemented concurrently where the visual and motor systems compete for execution. An arbitration mechanism selects the appropriate action, which in turn may depend on visual feedback available from the previous execution of a visual action.

2.4.1 Selective Attention

The camera signals contain far more information than what can be processed by a practical vision system. This creates the need for attentional mechanisms to allocate computational resources to relevant parts of an image. However, relevancy is not a static measure, but depends on the context and task of the agent. The visual routines theory suggests *shift of processing focus* as an elemental operation where the processing shifts to indexable locations (locations which are different from its surroundings in terms of shape, color, disparity, texture, etc.).

Selective attention is analogous to the way human eyes process information. Given a task such as visual search, the eyes perform quick jumps from one location of the scene to another, known as *saccades*, with short durations of *fixation* at a given location. The saccades occur through peripheral vision (low resolution) while fixations are done by foveal vision (high resolution). The extraction of visual information occurs during the fixations. Experimental results show that the eyes fixate on interesting and informative regions in the scene (Henderson, 2003). An example of saccades and fixations for a visual search task is given in Figure 2.8. Robot vision simulates this process by changing the resolution of the camera and/or by moving the camera. Attentional processing saves computational time and cost. It also performs data reduction by filtering the irrelevant parts of the scene.

In our work, we focus on *micro-saccades* around object parts rather than saccades across a scene. Based on the initial inference results, the system decides whether additional evidence is required, and selects the image regions on which further processing is needed. The object parts detected so far act as indicators of where to look for more evidence. Thus the system can selectively target the vision processing to specific regions of the image. An

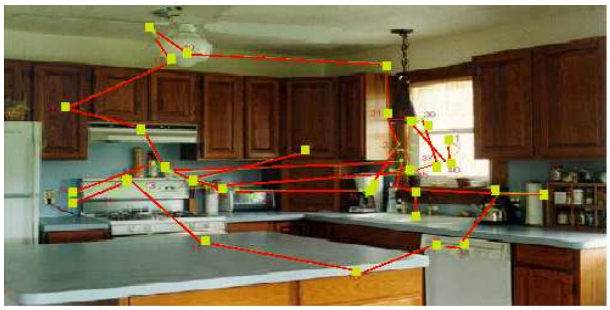


Figure 2.8: Saccades (red lines) and fixations (yellow squares) during visual search.

Figure 2.9: An example of active vision as deployed in the proposed work.

example of how active vision is deployed in our work is shown in Figure 2.9.

In the next section we give an overview of the related works on the topics discussed previously, for vision applications.

2.5 Related Work

The concept of visual routines was first proposed by Ullman (1984) as a way of explaining intermediate vision in human beings. The thesis suggests that visual perception for high level vision tasks can occur in different stages: early visual operators applied in parallel over the entire visual field forms the base representations, on which visual routines (assembled from elementary visual operators) can be applied to generate the incremental representations for the task at hand. This saves computational time and complexity and helps in selective processing.

Following this, several work has been done using visual routines in cognitively oriented tasks. (Agre and Chapman, 1987) and (Chapman, 1991), implements visual routines for automated game playing in a simulated environment (PENGI and Sonja). These methods use visual routines as a means to shift the processing focus or as attentional markers and

bypass the early vision processing, whereas the hard part of visual routines is in modeling the low-level vision processing and the uncertainty involved in their output. (Horswill, 1995) proposed an implementation of a visual routines processor for visual search on real camera images. But it does not model the uncertainty in the results of visual operators. Moreover, objects are identified using color cues and recognition is not addressed explicitly. Other work on visual routines are (Johnson, 1993) which assembles visual routines for hand detection using genetic programming, (Bala *et al.*, 1996) for eye detection using genetic algorithms and (Rao, 1998) which proposes a language of attention to generate visual routines. In this thesis, we address the problem of uncertainty and incompleteness in visual routines through an explicit object categorization task in an SRL framework. While most work on visual routines focus on visual attention, assembly and learning of routines, we present a formal approach for reasoning about them in the context of object categorization.

Research on the use of graphical models for vision applications focus on purely probabilistic generative and discriminative approaches. An application of Markov Random Fields (MRFs) for noisy object detection is proposed by (Cooper and Prokopowicz, 1991) where an MRF is constructed for a line detector which is functionally equivalent to its Hough transform parameter network. The image pixels are the nodes for the MRF and a MAP estimate is used to determine the most probable state of the pixels as to whether the pixels form part of a line or not.

With the advent of Conditional Random Fields (CRFs) (Lafferty *et al.*, 2001), several work has been done on image classification and segmentation using CRFs and are shown to outperform MRFs since they allow to relax the conditional independence assumption on observed data, giving them the ability to directly model the conditional probability. Researchers have employed many variants of CRFs for vision applications such as Discriminative Random Fields (Kumar and Herbert, 2003) and Tree-Structured CRFs (Awasthi *et al.*, 2007).

All of the above mentioned methods have a major drawback: they have a rigid structure and cannot represent variable number of objects and general relations between objects. The inference in these models are of a propositional nature and the models need to be replicated

across the image to detect multiple objects.

There has been few other approaches, apart from graphical models, that exploit the structure and interactions between objects in vision applications. (Sridhar *et al.*, 2008) and (Dubba *et al.*, 2010) use spatio-temporal relations between objects for learning functional object categories and event models respectively. Another work (Warden and Visser, 2011) performs spatio-temporal analysis on dynamic scenes to improve the grounding situation of autonomous agents in simulated physical domains. (Antanas *et al.*, 2010) investigates how simple logical generalization techniques can help in identifying known structures in images. (Ijsselmuiden and Stiefelhagen, 2010) proposes a temporal logic framework for high-level activity recognition from perceptual inputs. There has also been work on developing a visual grammar for object representations (Song-Chun and Mumford, 2005) and on representing objects using deformable parts (Felzenszwalb *et al.*, 2010).

An attempt at combining logic with uncertain reasoning is proposed by (Shanahan, 2005) which uses abductive inference for object detection along with an explanatory value attached with the hypothesis. The explanatory value is defined based on probability and is a measure of the truthfulness of the hypothesis. A recent work (Shet *et al.*, 2011) uses first-order logic to parse image features and to detect the presence of different patterns of interest for human detection and aerial object detection. It handles uncertainty in rules and observations using bi-lattice structures. The aim of their work is to detect different patterns of interest (object verification) and not object categorization. SRL models (or graphical models in general) encode the influence between variables directly whereas bi-lattices encode them weakly in the rules themselves. Other work on uncertain reasoning can be found at (Chachoua and Pacholczyk, 2002), (Mailis *et al.*, 2010), (Qin *et al.*, 2011) and (Weng and Chen, 2010).

Purely probabilistic approaches are not flexible to model complex environments and purely logical approaches cannot handle noise and uncertainty in a principled way. SRL models (Getoor and Taskar, 2007) provide an efficient inference mechanism by exploiting the relational structure of data and capturing generalizations among them, as discussed earlier. One such model is the Markov logic network which combines the power of first-

order logic to handle complex environments along with the advantages of probabilistic models. For a detailed overview of MLNs, please refer to the work by Domingos and Lowd (Domingos and Lowd, 2009).

There has been very few applications of MLNs in vision related tasks. One such application is in visual event modeling and recognition (Tran and Davis, 2008) where detected primitive events are grouped into composite events using probabilistic inference. The domain knowledge is encoded using Markov logic. A recent work on object detection in a home environment (Wu and Aghajan, 2010) employs user interactions on the objects as features for the MLN to detect the objects. A hierarchical activity analysis is performed using a camera network and the object-activity relationship is encoded in the MLN for detection. Another work on entity resolution in images (Chechetka *et al.*, 2010) uses Markov logic to represent the contextual information across images in a face recognition dataset. But the aim of their work is different from the work proposed in this thesis in that they deal with object-instance identification (associating faces with individuals across a database) while we are looking at the problem of object class identification. Object class identification is a harder problem as suitable features have to be incorporated which can generalise over objects within the same class while discriminating objects between different classes. To the best of our knowledge, our work is the first approach in applying MLNs as a layered architecture for object categorization using basic visual features such as shape and spatial relations. As far as we are aware, this is also the first work on using relational models in active vision.

2.6 Summary

In this chapter we discussed the theory of visual routines and how it can be used for visual perception of complex structures in a modular way. We presented an overview of the various methods for uncertain reasoning. We described graphical models as a tool for probabilistic inference and mentioned the drawbacks of traditional models and the need for relational models. We introduced Markov logic networks as a relational model and gave an

overview of belief propagation algorithm for inference. The active vision paradigm and its advantages were discussed and the concept of selective attention was described. We also gave an intuitive idea of the proposed work in relation to MLNs and active vision. The chapter concluded with an overview of the related work in these areas.

CHAPTER 3

PROPOSED FRAMEWORK

In this chapter, we describe the design and implementation of the proposed hierarchical MLN model and its integration with active vision. To the best of our knowledge, this is the first work employing Markov Logic Networks in a hierarchical fashion for any vision related tasks. The integration of the hierarchical model with active vision is also unique to our work. Initially we explain the general architecture and the major modules of the framework. Towards the end of this chapter, we describe the implementation of the system for the object categorization task.

3.1 Evolution of the Idea

The problem which we address in this thesis is on reasoning with uncertain and incomplete visual evidence. Our aim was to develop a robust system for object categorization, specifically for embodied agents. This entailed that the visual input to the agent would be noisy and have an uncertainty associated with the object category. But the agent could modify its initial guess by trying to obtain finer details of the object (*'look harder'*) through interaction with it. Thus, we needed to model

- Noisy operators: noise in the results of the visual operators
- Multiple shots at the same image: at different resolution, sensitivity, pose, etc.
- Objects of geometric regularities suitable for robotic simulation tasks

Bayesian modeling seemed to be a likely solution for the reasoning system. Initially we thought of modeling the system using MRFs as follows: detect basic visual features

from pixel information at the base level and built higher level MRFs over it in a hierarchical manner to model complex objects. But as explained in Chapter 2, traditional graphical models are cumbersome to model objects with multiple configurations and to extend to higher level structures due to their propositional nature of inference. The following problems had to be addressed: (1) how to incorporate top-down knowledge (composition of objects as lines, faces, etc.) in MRFs in an efficient way, (2) how to model getting another observation of an object part (i.e., the decision to look harder). We chose active perception as a method to obtain additional evidence since it was suitable to be deployed on embodied agents due to their ability to perceive as well as interact with their world. But we needed an efficient mechanism to detect multiple objects in parallel and handle the general configurations of the objects. Statistical relational models were then chosen as a possible method as they allowed for reasoning with uncertain evidence in a way similar to MRFs and could also handle general relations between objects using a single template. We chose Markov logic networks as the model because they provided a simple and intuitive representation of the object features using first-order logic rules. Hence it could be extended as a hierarchy to reason at multiple levels of uncertainty. Moreover, it provided efficient inference algorithms to detect objects in parallel. The object parts were then chosen as indicators of where to look harder.

3.2 General Architecture

In this section, we present our novel framework for reasoning with uncertain and incomplete evidence generated by visual routines, using multi-layer inference and selective visual processing. We develop a complete end-to-end system for the task of object categorization in embodied vision systems. The general architecture of our system is shown in Figure 3.1 and is divided into four key modules. We outline the key ideas behind the modules in this section and provide more implementation details in the next section.

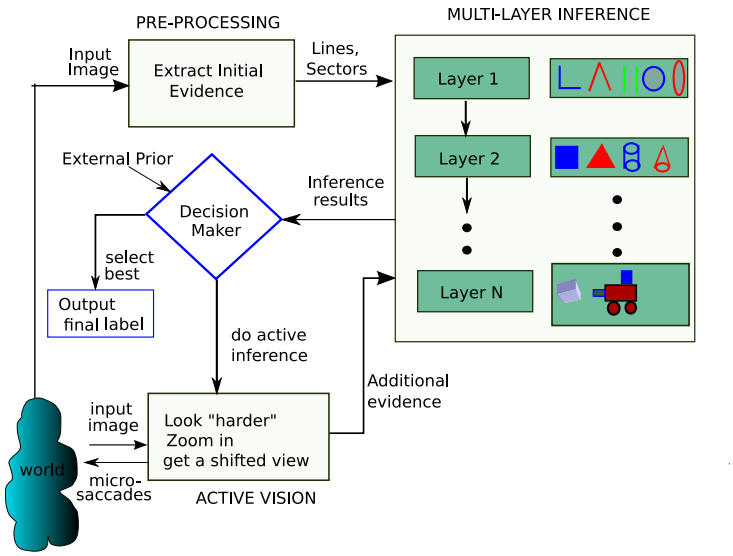


Figure 3.1: General architecture of the proposed framework.

3.2.1 Pre-processing

This is essentially the base layer (*layer 0*) of the inference module. It extracts the base evidence from the image, consisting of lines, sectors and corners using visual operators such as *Canny edge detector, Hough transform for lines and circles, corner detector*, and *contour extractor*. This evidence is used as input to the inference module to derive higher level features and objects from it. This layer forms the base representation on which high-level visual routines are applied.

3.2.2 Multi-layer Inference

This module performs the initial categorization of objects, using a set of MLNs organized in a hierarchy. The system contains pre-defined models of objects as a composition of object parts and their spatial relationships. The object category is inferred by matching the extracted features against these models. The inference of the object is performed across multiple layers, where the MLN at each layer runs *belief propagation* to perform inference on the parts of objects associated with that layer. The inferred object parts at lower layers, along with their locations and beliefs on their category, are propagated to the higher lay-

ers, which extracts higher level features from them and combines the object parts to infer complex structures. The MLNs at each layer runs independently and the supportive evidence from the lower layers is passed upwards through a well-defined procedure explained below. The abstraction at a layer is formally defined as follows:

Each layer has a set of nodes $\langle \mathbf{E}_i, \mathbf{O}_i \rangle$.

Input at layer i:

$$\mathbf{E}_i = \left\langle \mathbf{e}_i^{LL}, \mathbf{e}_i^{IL} \right\rangle \tag{3.1}$$

lower-layer evidence:

$$\mathbf{e}_i^{LL} = \left\{ \mathbf{e}_1^{LL}, \mathbf{e}_2^{LL}, ..., \mathbf{e}_{i-1}^{LL} \right\} \tag{3.2}$$

is the evidence propagated to layer *i* from lower layers.
in-layer evidence:

$$\mathbf{e}_i^{IL} = v_i(\mathbf{e}_i^{LL}) \tag{3.3}$$

is the set of features obtained by the application of visual routines associated with the layer, v_i, on the evidence propagated to the layer.

Output at layer i:

$$\mathbf{O}_i = \cup_j \langle O_{ij}, p_{ij} \rangle \tag{3.4}$$

is the set of query nodes at layer *i* (objects/object parts) along with their output probabilities.

Let *evd* (O_{ij}) be the subset of evidence available for inferring query *j* at layer *i*, such that

$$\mathbf{E}_i = \cup_j evd(O_{ij}). \tag{3.5}$$

The inference for node O_{ij} is given by the marginal probability P ($O_{ij}|evd(O_{ij})$)

$\mathbf{O}_i$ influences $\mathbf{E}_{i+1}$, via an interface function that maps the probability distribution over

O_i to a distribution over e_i^{LL} as given below:

$$e_i^{LL} = f_{i,i+1}(\mathbf{O}_i). \tag{3.6}$$

The interface function $f_{i,i+1}$ takes as input the set of query nodes and their associated probabilities and produces a set of evidence nodes for the next layer.

The multi-layer inference can be viewed as a "gated" belief propagation across the layers. Each layer runs a belief propagation and once the belief propagation at *layer i* has converged, some evidence is propagated from *layer i* to *layer i+1*, which is a function of *layer i* inference results and *layer i+1* query. In our work, a thresholding function along with a typecast operator is used to determine which predictions from the previous layer are to be used as input for the current layer. More details on these functions are given in Section 3.3.

3.2.3 Visual Routines

Following the work of (Ullman, 1984), we define visual routines at each layer of the hierarchy to extract the features associated with the layer. Each layer in the inference module outputs subsequent incremental representations generated by the application of high level visual routines associated with the layer. The input to a layer determines the specific locations on which visual routines are to be applied. These are essentially the locations of the supportive evidence selected by interface functions from lower layers. In logic terminology, the high level visual routines are the predicates of the FOL rules. A composition of visual routines through logical connectives is used to establish the features to be inferred at the layer.

As an example, consider the rule for inferring a cube at a layer, from L-face and *Square*-face as evidence:

$$\text{l-struct } (ls) \wedge \text{isSquare } (sq) \wedge \text{isTranslated } (ls,sq) \Leftrightarrow \text{cube } (ls,sq)$$

Here, the antecedent of the rule represents the composition of visual routines, while the consequent is the object part to be inferred. In this example, the visual routine *isTranslated ()* checks if the corresponding corners and faces of L and square are linearly shifted, to infer whether the shape property *cube* holds among the evidence at the layer. This high level visual routine is applied on the L and square faces alone, which are the evidence propagated from the lower layers. Thus we build a hierarchy of visual routines using a formal grammar (Markov logic). This helps in identifying the structure present in an image in a systematic way. The routines also help in generating structured representations of input and output between the layers.

The visual routines are designed as *soft* predicates. The output of the predicates (routines) is boolean but the decision (*true/false*) is made using a tolerance range instead of an exact match. For example, the routine *intersectsOrdered()* which checks for intersection of two lines, allows the line end-points to be within a certain radius from each other. Thus the uncertainty in the result of operation of visual routines are handled in two ways: (1) belief propagation in MLNs and (2) soft predicates that can tolerate a certain amount of imprecision.

At *layer 0*, the routines constitute the basic image processing functions used to extract *lines*, *sectors*, *etc.* which are applied in parallel over the image. At higher layers, the routines are the predicates used in the MLN rules. The routines at higher layers do not perform explicit image processing, but they operate on object parts propagated from its previous layers, which eventually have been generated by image processing at *layer 0*. The complete set of routines employed in our work are given in Chapter 4. Apart from these routines, we also use the *shift of processing focus* routine which is the active vision control used to *look harder* at certain regions of the image.

3.2.4 Decision Making

This module decides whether more information is required or to draw a conclusion on the object category. The decision is made based on the result of inference and the prior

(background) knowledge of the presence or location of objects. Based on the current belief on the object category and the knowledge, selective visual processing is applied to specific regions of the input, to gather missing evidence at *layer 0*, i.e., *lines, sectors,etc.*

3.2.5 Active Vision

This module helps in focusing on certain regions to gather missing evidence which could potentially improve the certainty of the object category being inferred. It is analogous to the *shift of processing focus* operator described by Ullman. Ithelps in making the inference tractable because it removes the irrelevant predicates which would otherwise need to be inferred if the detection was done jointly over all the predicates in the domain. Since this module provides the option of focusing closely on regions of the image that we are actually interested in, we can carry out the initial visual processing with low cost operators and with strict thresholds to remove spurious/irrelevant evidence. Based on the object parts inferred in the initial run, the "interesting" regions can be processed at a relaxed threshold or with complex operators to yield the finer details of the object.

Additional information can be obtained through various methods such as relaxing the thresholds of visual operators, zooming in on a part of the object or by taking another snapshot of the object from a different view point. Essentially, this can be understood as the agent "looking harder" at certain parts of an object for more evidence, given some initial evidence.

The initial inferencing is performed in a bottom-up manner until *layer N* (the maximum possible layer of inference according to the given data) is reached, where a decision on whether to "look harder" is made. The control then flows back to *layer 0* where additional evidence is obtained. The additional evidence is combined with the original evidence and the inference is run again from *layer 1* onwards. Thus a hybrid of bottom-up and top-down control strategy is followed between the layers until final inference is made. In our framework, active vision can also be viewed as trying to justify the prior knowledge on the presence of an object, by gathering more evidence about the object. Figure 3.2 shows the

various stages in our framework as opposed to single-stage processing.

3.3 Implementation

3.3.1 Pre-processing Module

The image processing functions are implemented in this module using the OpenCV (Bradski, 2000) library. The functions used are image smoothing, dilation, edge detection, corner detection, Hough transform, and contour detection. These functions form the basic visual operators which generate the base representations upon which high level visual routines are applied. The line segments are extracted using Canny edge detector and probabilistic Hough transform. Ellipses are extracted by detecting contours on the image and fitting the contour points onto an ellipse. Circles are extracted using Hough transform. Corner points are extracted by the following procedure: An eigen value corner detector and a contour extractor are applied on the image. For each contour, we select those corner points which fall within a particular radius of the contour centre. Each contour is given a score $S = \frac{variance}{numOfSelectedCorners}$ based on the variance and density of corner points around its centre. The optimized corner points are then selected by thresholding based on their proximity to contour centres and the scores and moments associated with those contours. We use two sets of parameter values for the image processing functions: the first is a set of strict thresholds for the initial processing of the image and the second is a set of relaxed thresholds for active vision. This module returns a set of locations of the extracted lines, sectors and corner points.

3.3.2 Inference Module

Markov Logic: We chose MLNs for the underlying inference module since it provides a relatively simple and intuitive way for representing the features, domain knowledge and inference results based on first-order logic. The first-order rules form a single template

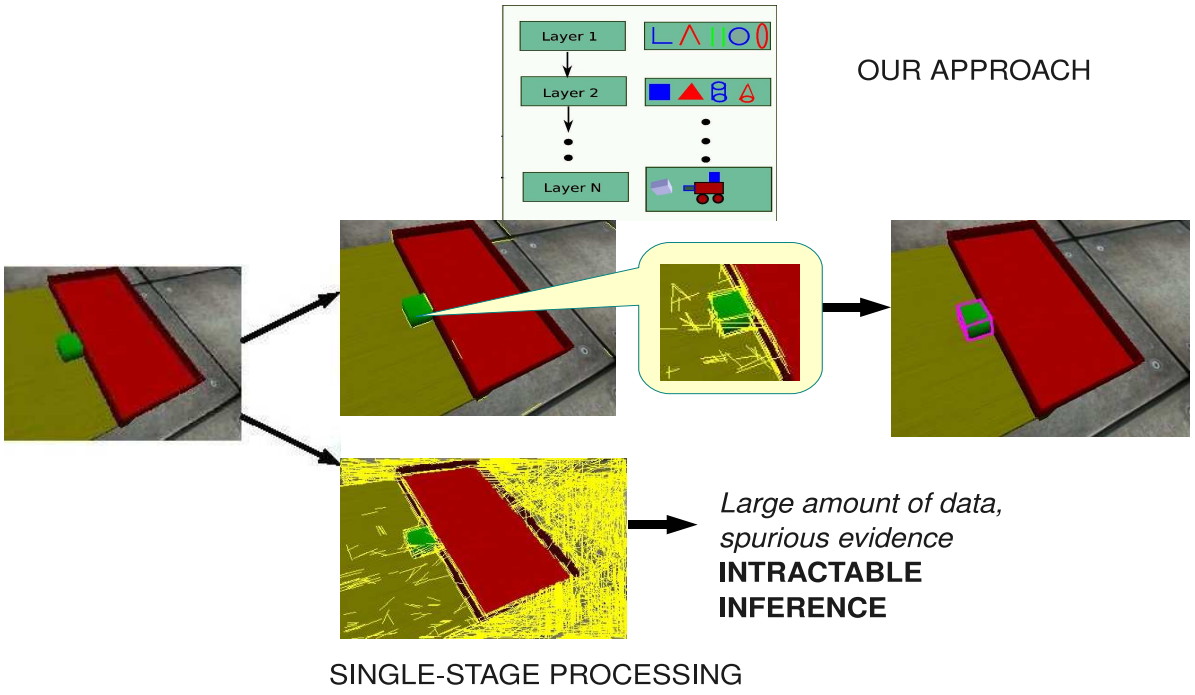


Figure 3.2: Our framework as opposed to single-stage vision processing. In the initial stage of vision processing, strict thresholds are used which give just enough lines (marked in green) on the cube to generate object parts to look harder at. Active vision is done on the selected region with relaxed thresholds. Final inference result is obtained combining the new evidence with the original evidence. The data required if entire processing is done using a single stage is presented in the lower half of the figure. As can be seen, large amount of spurious data is generated, possibly making the inference intractable.

for all possible configurations of the object. In our work, the predicates are the visual routines which will check for the corresponding properties in the input image. When viewed as a factor graph, these predicates act as variable nodes and the rules as factor nodes. Those predicates which appear together in a rule are connected to the same factor node. Belief propagation is used for inferring the distribution over the query predicates. When a predicate has many arguments, the memory requirements for processing grows rapidly. In order to keep the inference process tractable, we limited the number of object parts combined to two at a time in each layer. With a single MLN for an object type O_{ij} we can infer the presence and location of object types of different configurations across the image.

Features: We use two classes of features: *shape* properties ($\mathbf{e}_i^{LL}$) and *spatial* relations ($\mathbf{e}_i^{IL}$). The shape properties are propagated as evidence across the layers. The shape features at a layer are essentially the object parts inferred and propagated from its lower layers. The spatial relations are extracted at each layer from the shape features and are not propagated between the layers explicitly. But the spatial features get embedded in the representation of object parts inferred at each layer.

Interface function: The interface function $f_{i,i+1}$ is composed of thresholding function, normalization and a type cast operator. We threshold the inference results at a layer based on their probability, in order to avoid too many combinations of object parts being inferred at the higher layer. The thresholding operator for an output predicate (*O*,*p*) is given by:

$$(p \geq thresh) \wedge (thresh \geq 0.5). \tag{3.7}$$

where

$$thresh = 0.75 * (\max_j p_{ij} - \frac{\sum_j p_{ij}}{\sum_j 1}). \tag{3.8}$$

In essence, this corresponds to propagating the beliefs that fall within the upper quartile.

The evidence at each layer is represented as a set of weighted facts. Hence, the probabilities associated with the propagated evidence has to be converted to weights in the Markov logic network at the higher layer. We use min-max normalization to convert the probabilities in the range [0 1] to weights in the range [-w w]. The value for *w* is chosen empirically as the minimum weight needed to obtain the same inference results when weights are set for evidence of probability 1 as compared to unweighted evidence. (Note that according to the convention in Alchemy, unweighted evidence is considered as sure evidence.) The value of 16 was found to be suitable for our application.

The shape properties at each layer are specified as typed constants. The object parts which are selected from the lower layer are converted to their appropriate type and are passed on to the subsequent layer. The type cast operator provides two advantages: (1) It helps in restricting the number of arguments per predicate to a maximum of two, making the inference tractable. (2) Since the types for query and evidence nodes of a rule have to be compatible, it eliminates a lot of irrelevant nodes which would otherwise have been part of the belief propagation network, thus producing savings in time and memory requirement. The evidence from the lower layers are also propagated till the final layer of inference for the object. The relative orientations of the object parts inferred at each layer are maintained inherently by the predicates at the layer.

As an example, consider identifying a square

Layer 1:

isLine (l_1)$\wedge$ isLine (l_2)$\wedge$ isPerpendicular (l_1, l_2)$\wedge$ intersectsOrdered (l_1, l_2)
$\wedge$ suitableSize (l_1, l_2)$\Leftrightarrow$ l-structure (l_1, l_2)

Layer 2:

l-struct (ls_1)$\wedge$ l-struct (ls_2)$\wedge$ intersects (ls_1, ls_2) $\Leftrightarrow$ square (ls_1, ls_2)

The presence of a square is inferred in two layers. *Layer 1* infers the presence of *L*-structures from the base evidence where the features checked for are the perpendicularity of the combining lines, suitable length of lines and their intersection. The *l-structures*

inferred from *layer 1* are type cast as *l-structs* and are passed on to the next layer, weighted by the probability associated with their label. *Layer 2* infers the presence of squares from the evidence received by it from the lower layer. The features derived at this layer are the intersection properties of *l-structures* to infer the presence of square in the image.

The layered inference helps in a more accurate categorization of objects, since the object parts itself are evaluated for existence at lower layers. This allows for pruning of the category of objects/object parts to be inferred at the subsequent layers. Pruning is extremely important as an inference over all the possible predicates (both simple structures and composite structures) would make the inference highly intractable. Layering also helps in performing active vision and inference on selected parts of the object, thereby reducing the cost of "looking harder" at the entire object/scene. The positional information of object parts are available from the inference results of lower layers, which may be of use where the vision system is integrated with the motor system of the robot for further tasks (eg., the positional information of the handle of a cup helps in grasping it). It also provides the common advantages of any hierarchical system, such as modularity and reuse of the lower layers to model different higher level objects from the same base parts.

3.3.3 Decision Making Module

After the initial run of multi-layer inference, control reaches this module which decides whether additional visual processing should be performed or whether the object category can be concluded. This module is driven by the prior knowledge and the results of inference. In our work this knowledge is essentially the presence of a particular category or presence of objects at a particular quadrant (top-left, top-right, bottom-left, bottom-right) or both and is domain-specific. For eg., if the query is "Is there any object in the scene?", the knowledge would be lesser as compared to "Pick up the cubes in the scene". The latter problem is more specific and provide better information in that it is clear that there are some cubes in the scene and at least one of them has to be identified by the algorithm. The prior knowledge on location is provided as a probability value for each of the four

quadrants and the prior on category is given by specifying the class name of the object to be searched for. By default, uniform prior is assumed for the locations and categories. The prior knowledge could be set depending on the given query. This module also targets the visual processing to specific parts of the image. The module decides to call active vision in the following cases:

- There are object parts inferred from lower layers, which could not be described as part of a complete object, and there is some prior knowledge on the presence of objects of a certain category and/or at a specific region.
- The object parts are successfully explained as part of whole objects, but there is enough background knowledge that forces looking for objects of a particular category and/or at a specific region.

In the first case, a small region of interest around the object part is examined for more evidence. In the second case, if the quadrant to be searched is not available as prior knowledge, the entire image has to be re-examined. Gathering additional evidence helps in improving the quality of inference.

3.3.4 Active Vision Module

This module gathers additional evidence about objects by calling the pre-processing module with relaxed parameters for the visual operators. The image on which active vision is performed is a cropped region of the original image. The specific region to be processed is decided by the decision making module. Other methods by which active vision can be performed are zooming in on a part of the scene or getting a different view of the object with the help of a camera. Currently, we use static images and hence tune the visual operators at *layer 0* ("look harder") to detect lines and sectors at a lower granularity.

Active vision essentially changes the belief of shapes as being object parts. This is achieved by tuning the visual operators at the lowest layer. But it indirectly affects the inference at higher layers due to the bottom-up nature of the inference. Of course, it is possible to tune the visual operators at higher layers by relaxing the parameters of the high-level visual routines, but this is quite hard and requires extensive engineering. Active

Table 3.1: An example set of clauses used to identify a train shape.

Layer	Algorithm
L1	isLine(l_1)∧ isLine(l_2)∧ isPerpendicular(l_1, l_2)∧ intersectsOrdered(l_1, l_2)∧ suitableSize(l_1, l_2) ⇔ l-structure(l_1, l_2) isSector(s_1)∧isSector(s_2)∧isParallelSectors(s_1, s_2)∧isDifferentSectors(s_1, s_2)∧ suitableAsCylinderSectors(s_1, s_2)∧suitableDistBwSectors(s_1, s_2)⇔cylinderSectors(s_1, s_2)
L2	l-struct(ls_1)∧ l-struct(ls_2)∧ intersects(ls_1, ls_2) ⇔ square(ls_1, ls_2)
L3	cubeface(f)∧isSquare(f)∧cylinderSectors(cys)∧abutsWheel(f,cys)⇔trainBody(f,cys)
L4	cubeface(f)∧isSquare(f)∧tbody(tb)∧abutsFront(tb,f)⇔trainBodyFront(tb,f)
L5	cubeface(f)∧isSquare(f)∧tbfront(tbf)∧abutsTop(tbf,f)⇔train(tbf,f)

vision can also be viewed as justifying the prior knowledge about an object category by gathering more supportive evidence. An example for active vision processing can be found in Figure 3.2.

Example: The MLN clauses used at various layers to infer a train object is presented in Table A.3. As can be seen from the table, the rules at the lowest levels are the ones corresponding to primitive shapes such as l-structures and wheels. The next level is about a square, the third level is about a trainbody, the fourth is about the front of the body and the final level reasons about the train. As can be seen, the rules are progressively used to infer objects at higher levels. We do not present the background (prior) knowledge here. These are just the MLN clauses used for inference of the train object. An example figure for detection of train shape is given in Figure 4.4.

3.4 Summary

In this chapter we presented an SRL approach for reasoning with incomplete and uncertain visual evidence. The implementation of the system for an object categorization task for embodied agents was provided. The result of operation of the visual routines could be unreliable and noisy due to the limitations of the camera as well as the vision processing methods. The hierarchical MLN model was proposed which could reason about object parts at different levels and under varying levels of uncertainty. The framework provides

the capability to build complex object structures as a hierarchy in a systematic way, using basic visual routines. The proposed model can recognize multiple objects across the image using a single template. It uses simple, intuitive rules to represent the features which are obtained by the application of visual routines at the respective layers. The hierarchical model helps in selective pruning of object classes, as we need not reason about objects which do not have sufficient supportive evidence from the lower layers.

Our work is cognitively motivated and provides a way of integrating the active vision paradigm with the hierarchical model. The active vision techniques help in achieving a robust and reliable detection of objects. It is advantageous to be deployed in embodied vision due to their inherent interaction with the environment. It also helps in filtering irrelevant data thus producing efficient memory and time requirements due to the selective processing. The combination of a hierarchical model with active vision helps in selective pruning of object classes, as we need not reason about objects which do not have sufficient supportive evidence from the lower layers.

The implementation details describe the Markov logic rules and predicates used at each layer. The basic visual features used and the interface function between the layers are also discussed. The rules are designed based on simple geometry of the objects. The hierarchical model provides a design for object categorization from incomplete evidence using basic visual features. The experimental evaluation of the proposed system is discussed in the next chapter.

CHAPTER 4

EXPERIMENTAL EVALUATION

This chapter discusses the experimental validation of our approach. Brief description on the datasets chosen, parameter settings, results and implementation details are provided. The comparison of the performance of our system for various levels of incomplete and noisy evidence is reported for synthetic images. The system is further evaluated on an embodied agent and also on real images. Comparison of various algorithms implemented in MLN is provided. A discussion on the use of texture operator as a feature and related experiments are provided towards the end of this chapter. The chapter concludes with comparison of our system with a baseline texture detector.

4.1 Experimental Setup

We use the Alchemy toolbox (Kok *et al.*, 2007) for implementation of the MLN inference. Currently we have modeled seven classes: square, triangle, cylinder, cone, sphere, cube and train. Belief Propagation (BP) is used for inference as it was found to work better than other inference methods in MLN. We compared the performance of BP and another state-of-the-art sampling algorithm in MLNs, namely MC-SAT (Poon and Domingos, 2006), through the various experiments explained in the section on synthetic images. Since the domain mainly consists of Horn clauses (we convert the double implication to a pair of single implications), it appears that BP was more accurate in the predictions while MC-SAT is still a sampling method and hence misses a lot of true positives and introduces false positives. Since the ground Markov network does not have too many cycles, BP converges to the true distributions. In our experiments, MC-SAT gave a lot of false positives on examples where the labels could be potentially confused.

4.1.1 Datasets

We use three different sets of images for experimental validation:

- **Synthetic images:** This set consists of images generated using the OpenCV library and hand-drawn images. The images generated by OpenCV consists of objects of random category and position. The images are corrupted with RGB and white noise for the purpose of evaluation. The hand-drawn images have staggering object boundaries which makes extraction of features hard and thereby provide noisy and incomplete evidence.
- **iCub images:** The iCub (Metta *et al.*, 2008) is a humanoid robot developed for studies on cognition. We use the iCub simulator to validate our framework on an embodied agent. The simulator accurately replicates the physics and dynamics of the real environment and is a popular tool for simulation studies of robotic tasks. The iCub images are obtained by systematically moving the iCub simulator head over objects placed on a table and recording the images from the camera mounted on the iCub simulator head.
- **Real images:** This set consists of images of a Rubik's cube obtained from Microsoft's Kinect Xbox (R), by rotating the Kinect in fixed steps of five degrees. In these images, the faces of the cube are not uniformly colored and the images obtained have significant variation in illumination due to the rotation of the Kinect. These factors makes the task of detection and vision processing complicated.

Detailed explanation of the images used in each dataset are given along with the description of the experiments conducted with the respective sets. The object categories considered for synthetic images are square, triangle, cylinder and cone, while the categories considered for iCub and Kinect images are cubes and spheres. The details of the experiments with these sets are explained in the following sections.

4.1.2 Parameters

The parameters to be tuned in our system are the vision thresholds, the thresholds for the spatial features, and the weights used in the MLN rules.

The vision parameters are Canny thresholds, Hough thresholds, corner detection thresholds, level of smoothing and zoom. Table 4.1 shows the parameter settings for initial and

Table 4.1: Parameter settings of the visual operators.

Parameter	Synthetic images		iCub images		Real images	
	Initial Processing	Active Vision	Initial Processing	Active Vision	Initial Processing	Active Vision
Canny low threshold	10	50	80	80	80	80
Canny high threshold	70	100	200	200	200	200
Canny aperture	3	5	3	3	3	3
Hough line accumulator threshold	120	120	10	10	10	10
Min Hough line length	50	50	10	10	10	10
Hough line link threshold	5	5	7	25	7	7
Num of Dilations	1	1	1	1	1	1
Corner quality level	-	-	-	0.007	-	0.03
Eigen block size	-	-	-	3	-	3
Min. distance b/w corners	-	-	-	7	-	7
Gaussian smoothing kernel size	9	9	9	9	9	9
Zoom level	1	1	1	1	1	1

active vision processing, for the three datasets used. Note that for iCub and Kinect images, the corner detector is used for gathering evidence in the active vision stage.

The predicates are designed based on simple geometry of the objects. The spatial predicates are soft predicates which allow for some tolerance of noise, i.e., intersection of `L´s need not be an exact intersection, but a tolerance range is used. The thresholds for all the predicates were set empirically, based on the true and false positives obtained in a validation data set. Table 4.2 shows the spatial predicates used in our system. The shape predicates are the query predicates at different layers and are presented in bold in the table.

The prior knowledge can be specified with respect to labels and/or regions (top-left, top-right, bottom-left, bottom-right quadrants). Recall that the prior knowledge is essentially an external bias and we use it in a heuristic fashion. In the absence of any knowledge, all the inferred labels and regions are used for the next layer. The results presented in the thesis do not assume any prior knowledge.

Table 4.2: Spatial and Shape predicates used at each layer. The shape predicates are in bold. The shapes at lower layers are reused to form parts of the different higher level shapes

Layer	Spatial and Shape Predicates
L1	isPerpendicular(line, line), intersectsOrdered(line, line), isDiff(line, line), positiveIntersection(line, line), suitableLength(line, line), angleGtThanFive(line, line), isParallelLines(line, line), similarLength(line, line), suitableDistBwLines(line, line), isParallelSectors(sector, sector), isDifferentSectors(sector, sector), isRound(sector), suitableSize(sector), fullCurve(sector), suitableAsCylinderSectors(sector, sector) sufficientGap(corner, corner), **l-structure(line,line)**, **triangletwosides(line, line)**, **sphere(sector)**, **cyl-lines(line, line)**, **cyl-sectors(sector, sector)**, **cornerpair(corner, corner)**
L2	intersects(lstruct, lstruct), closedThreeSides(triangletwosides, line), closedObj(triangletwosides, sector), formsCylinder(cyl-lines, cyl-sectors) isDiffCpairs(cpair, cpair), isPerpendicularCpairs(cpair, cpair), suitablelengthCpairs(cpair, cpair), intersectsOrderedCpairs(cpair, cpair), **square(lstruct, lstruct)**, **triangle(triangletwosides, line)**, **cone(triangletwosides, sector)**, **cylinder(cyl-lines, cyl-sectors)**, **cpairlstructure(cpair, cpair)**
L3	isSquare(cubeface), abutsWheel(cubeface, cyl-sectors) cpairL-intersects(cpair-lstruct, cpair-lstruct), **squarecorners(cpair-lstruct, cpair-lstruct)**
L4	abutsFront(trainbody, cubeface), isTranslated(squarecorners, lstruct), **cube(squarecorners, lstruct)**,
L5	abutsTop(trainbodyfront, cubeface), **train(trainbodyfront, cubeface)**

4.1.3 Weight learning

Initially, the weights of the rules were set empirically according to the following heuristic:

$$P(y_i = 1) > P(y_j = 1) \forall y_i \in TP, y_j \in FP. \tag{4.1}$$

, i.e., the probability of the labels for the true positives is significantly higher than those for the false positives. By performing a line search on the space of weights between 0 and 1, we determined 0.7 to be the optimal value for the weight. We do not use hard constraints in our rules. The MLN gives a probability value for each combination of parts (shapes) that can potentially form an object queried for. The probabilities are thresholded and a MAP estimate is performed over labels on the same region of interest, to infer the final category.

We also learned the weights for the rules using the scaled conjugate gradient (SCG)

Table 4.3: Comparison of performance with learned and hand-coded weights at 1% salt-and-pepper noise.

Parameter	Without active vision				With active vision			
	hand-coded	SCG	DN	VP	hand-coded	SCG	DN	VP
Precision	1	1	1	-	1	1	1	-
Recall	0.26	0.26	0.13	0	0.39	0.42	0.28	0

Table 4.4: Comparison of performance with learned and hand-coded weights at 0.8% RGB noise.

Parameter	Without active vision				With active vision			
	hand-coded	SCG	DN	VP	hand-coded	SCG	DN	VP
Precision	0.98	0.99	0.98	-	0.96	0.97	0.93	0.94
Recall	0.34	0.35	0.22	0	0.47	0.47	0.33	0.09

method implemented in Alchemy. Each class has a set of features defined by the predicates in first-order logic. The weights for rules of each class is learned separately. For the training data, the positive and negative examples were taken in the ratio 1:2 for each predicate of a given class. They are labeled as positive or negative based on the predicate thresholds. The positive and negative examples are selected such that they fall slightly above or below the predicate threshold ($\pm$ 5 respectively). We observed that the training data constructed in this way was sufficient to learn suitable weights for the rules. For testing, we chose 95 synthetic images consisting of 190 objects of the various classes with two objects per image. The objects are in one of the following degrees of rotation: 0, 30, 45, 60, 90 with approximately forty objects per rotation angle. The images were treated with different levels of noise. Table 4.3 and Table 4.4 compare the performance of the system for learned and hand-coded weights at 1% salt and pepper noise and 0.8 % RGB noise respectively. More results on noisy images are included in the next section.

We observe that the learned weights are as good as the hand-coded weights. The tables also show the comparison of SCG with two other weight learning methods implemented in Alchemy: diagonal Newton (DN) and voted perceptron (VP). SCG weight learning

significantly outperforms the other two methods in recall. More details on these learning methods can be found at (Lowd and Domingos, 2007).

4.2 Evaluation

We empirically validate our system on objects with geometric regularities. Our framework can incorporate real-world objects that are not necessarily regular, if suitable visual properties such as color, histogram-based descriptors etc., are included as features and the MLN predicates are modified to handle grouping of non-geometric features. Currently, we use geometric shapes, texture and their relationships as the primary features, which leads to this restricted domain. We tested the framework on synthetic, iCub, and real images. We also compare the performance of the proposed hierarchical system with a baseline detector comprising of a single level MLN, for real images. The metrics used for evaluation are precision, recall and detection accuracy. The running time (time per image) on these datasets are $\sim$ 12-15 sec for synthetic images (1% salt-and-pepper noise) and $\sim$ 30-40 sec for iCub and real images. We assume that better processing times can be achieved with an optimized implementation of the system and evaluate the system for precision, recall and accuracy.

As described earlier, the proposed system models the uncertainty in visual routines by combining active vision and inference in a hierarchical MLN framework. Since our work is unique, we provide comparisons for the different components of our system through the experiments described in the later sections, but we could not find a complete end-to-end system which could be a fair comparison to the proposed system. The experiments on all these sets use the learned weights from SCG as described in the previous section. The details of these datasets and the experiments done on them are explained in the following sections.

4.3 Synthetic Images

This set consists of 95 images comprising of 190 objects with two objects per class. The objects are generated with random position and category and consists of the following classes: square, triangle, cone and cylinder. Each object is in one of the following degrees of rotation: 0, 30, 45, 60, 90 with approximately forty objects per rotation angle. The dataset is generated using OpenCV. Apart from this set, we also tested on hand-drawn images treated with 0.8 % RGB noise as shown in Figure 4.3. In order to test the effectiveness of our framework on incomplete and noisy data, we conducted the following experiments:

4.3.1 Incompleteness

We tested the framework on various levels of incomplete evidence. For this experiment, we bypassed intial image processing and assumed that all of the visual evidence is available initially. We removed some percentage of this evidence randomly to generate the input evidence. The test set consisted of 190 objects of different classes, but without noise and rotations. Figure 4.1 compares the detection accuracy (recall) at various levels of incompleteness and with active vision enabled/disabled. The figure also compares the performance of belief propagation and MC-SAT algorithms.

As seen in the figure, our framework can handle incomplete evidence effectively and depicts the power of the two main features of our framework: layered inference and active vision. We are able to focus precisely on the necessary regions due to the fact that we employ MLNs. The object parts detected at lower layers are indicators of where to look for missing evidence. The active vision component helps in obtaining the required missing evidence and enables fair detection even with a high percentage of incomplete evidence. Note that the use of active vision significantly improves the recall for both the algorithms. It is also observed that BP has a consistently higher recall compared to MC-SAT for both cases.

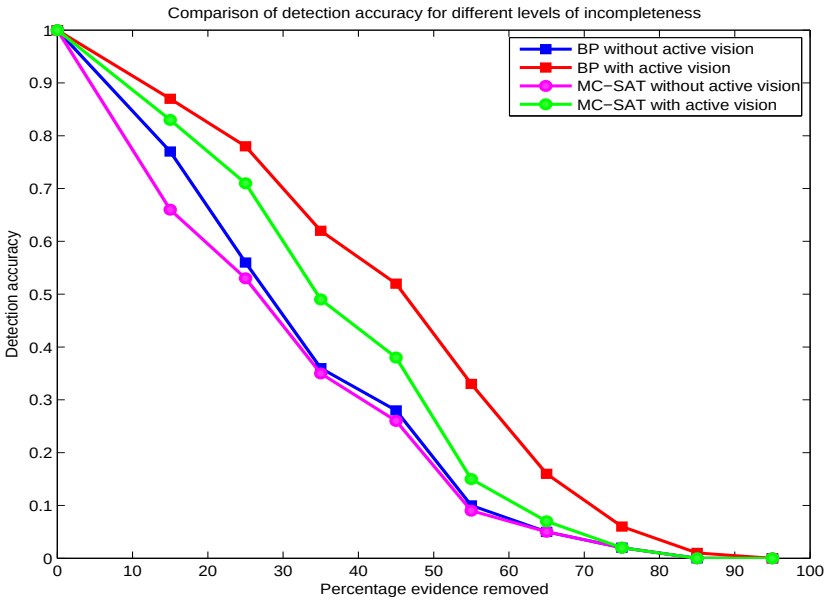
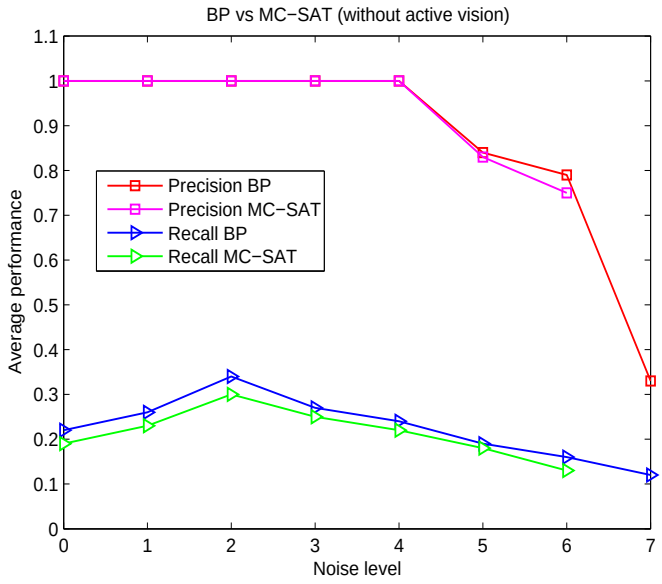


Figure 4.1: Comparison of detection accuracy at various levels of incomplete evidence.

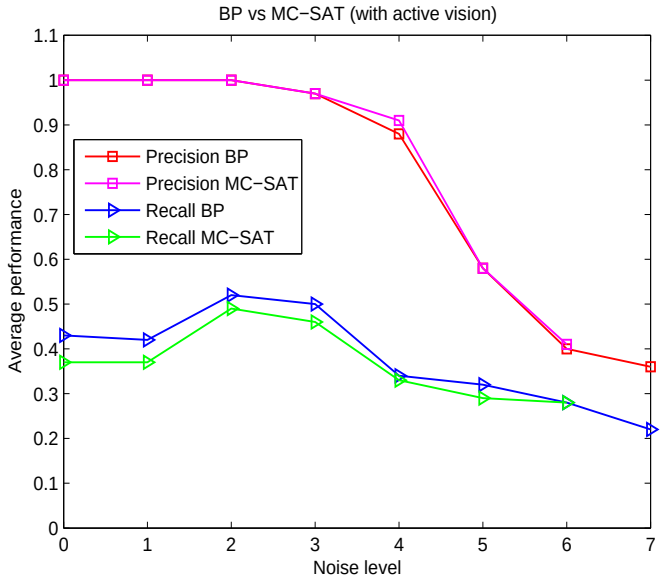
4.3.2 Noise

The test images consisted of 190 objects where the object could be in one of the five degrees of rotation described previously. The images were treated with different levels of salt and pepper noise. Figure 4.2 compares the precision and recall for BP and MC-SAT at different noise levels with active vision enabled/disabled. In this experiment, the visual evidence is obtained from actual image processing and initial image processing is not bypassed as done in our experiments with incomplete evidence.

The results prove the capability of our framework to handle noisy and inexact data and again emphasize the role played by the inference and active vision components. There is a decrease in precision when active vision is enabled as compared with inference without active vision. This is because the false evidence (and hence false positives) increases when processing is done at a higher granularity, due to the presence of noise in the images. There is gradual decrease in recall values with increase in noise level, which shows that our system is fairly resilient to noisy evidence. It is also observed that the recall improves in the initial few noise levels as compared to zero noise. This is because the presence of

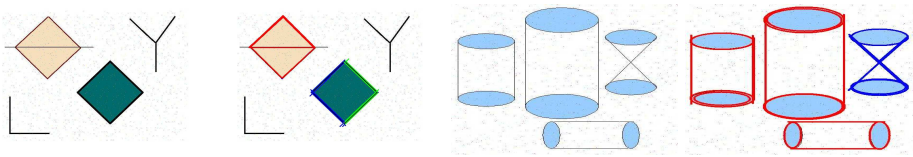


(a)



(b)

Figure 4.2: Comparison of performance at different noise levels: (a) Active vision disabled (b) Active vision enabled. The performance of MC-SAT at 7% noise level is not shown since it was took an unreasonably long time for execution as compared to BP.



(a) (b) (c) (d)

Figure 4.3: (a)-(b): Detection of squares and triangles on noisy image. The lines comprising the L's are shown in blue and green colors.The triangles are superimposed on the square on the top-left. The image does not highlight the object parts which did not become part of a final object. (c)-(d) Detection of cylinders and cones.

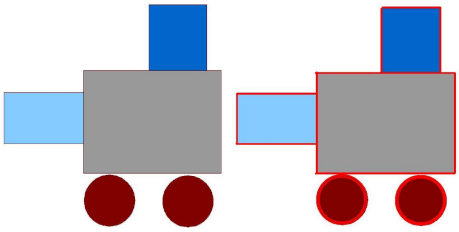
false evidence near the true edges causes the active vision to focus on those regions and thus helps in identifying the objects.

As explained previously, we chose BP as the inference algorithm since it significantly outperforms MC-SAT in precision and recall as shown by the experiments in this section.

Figure 4.3 shows the detection results for various classes on hand-drawn synthetic images. The images are corrupted by 0.8% RGB noise. As can be seen, our system performs accurately on synthetic images.

4.3.3 Complex Structures

We empirically show that our system can systematically build objects of complex structures, as shown in Figure 4.4. The layer-wise clauses used to infer the train shape is given in Table A.3. A major advantage is the tractable inference across multiple layers as compared to inference over a single MLN. Moreover, missing information can be gathered on object parts at any of the layers of the hierarchy. This makes the detection more robust and reliable as we have more accurate information on the regions to be considered for active vision because the category of the object becomes more defined as the inference progresses up the hierarchy. The hierarchy of object parts enables grouping and reuse of objects detected from lower layers into multiple categories at higher layers. For example, the square



(a) (b)

Figure 4.4: Detection of a train from basic shapes. (a) Input (b) Detection results marked in red. Active vision was not employed in this image.

object could be part of a train or a cube, parallel sectors could be part of a cylinder or the wheels of a train, and so on. The hierarchical inference also allows for object class pruning based on the evidence obtained from lower layers. More examples of detecting 3D objects such as cubes are given in the following sections.

4.3.4 Common Visual Routines for Multiple Objects

Our system has the ability to re-use visual routines for detecting multiple objects. Due to the layered architecture for the object model, basic shapes at lower layers become part of different complex objects at higher layers. Thus the set of visual routines required for detection of objects overlap. Moreover, the visual routines employed in our system check for geometric constraints over object parts. Hence, the same routines can be used for different objects with similar geometric features. Figure 4.5 shows the set of overlapping routines at any layer for the object classes of the synthetic image dataset. The figure also shows the relative degree of re-use of the routines. For example *intersectsOrdered()* has the maximum re-use, being applied for detection of *square*, *triangle*, *cone* and *train*. Due to layering, the set of routines used to detect the two sides of a triangle (*triangleTwoSides*) are also used in detecting the sides of a cone. Similarly, the set of routines used to detect *l-structure*, *square* and *cyl-sectors* are applied for the *train* object also to detect its body and wheels. The complete set of rules for the different classes are given in Appendix A. The

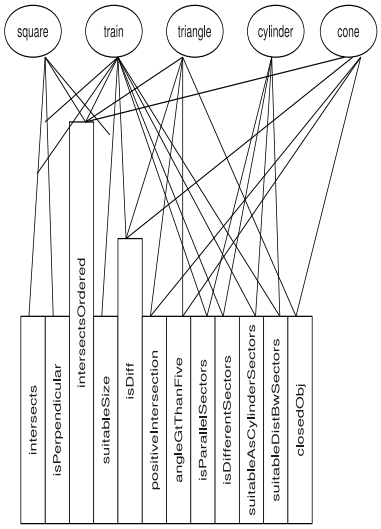


Figure 4.5: Overlapping visual routines for different classes of the synthetic datset. The figure shows the relative degree of re-use of the routines. The lines connecting the routines and classes indicate which classes employ the same routine.

routines also overlap for object parts within a layer as in *intersectsOrdered()* (*l-structure* and *triangletwoSides*), *isDiff()* (and), *closedObj()* (*triangle* and *cone*).

4.4 iCub Images

In order to validate our framework on an embodied agent, we tested our system on the iCub humanoid simulator. The iCub robot (Metta *et al.*, 2008) is a humanoid developed for conducting studies on cognition. The simulator accurately replicates the working of the iCub as well as the physics and dynamics of the robot's environment. More details on the simulator can be found at (Tikhanoff *et al.*, 2008). This experiment is part of a collaborative work [1] aimed at concurrent execution of the robot's gaze and motor systems through visually guided control.

[1] This experiment is part of a collaborative work done with University of Birmingham. The algorithms for the co-ordination of gaze and motor systems of the robot were developed by researchers at University of Birmingham, while the object detection part, developed by us, is explained in this thesis.

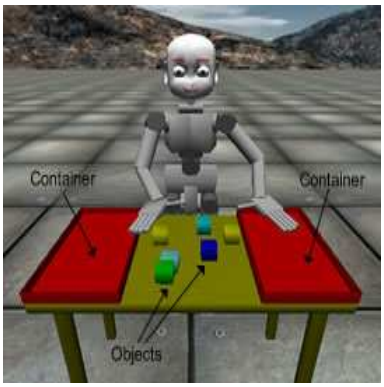


Figure 4.6: Table-top setting of the iCub humanoid simulator. The task is to clear the objects from the table and to place them in the containers present on the sides of the table. Figure taken from (Nunez-Varela *et al.*, 2012).

4.4.1 The Task

The task is to clear the objects from the table and to place them in the containers present on the sides of the table. The setting of the simulator for this experiment is depicted in Figure 4.6. The perceptual actions are executed concurrently with the physical actions of the robot. The robot has to decide where to direct its visual system and how to process the visual information while the other actions are being executed. This involves object recognition, gaze control to determine fixation points, motor actions (hands, in this case) for pickup/put-down objects and a decision making process for concurrent execution of the gaze and motor systems. The visual processing task here is to categorize the objects found in the region, for which a model for reasoning with uncertain visual evidence for embodied agents is proposed in this thesis. The algorithms developed for the co-ordination of gaze and motor systems of the robot can be found at (Nunez-Varela *et al.*, 2012).

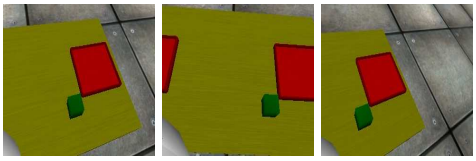
4.4.2 Results

For this task, we tested our system to detect 3D objects on a finely textured table-top setting. The object classes considered are sphere and cube. This dataset consists of 108 objects per class, with each object having a front, left and right view. An example for the

Table 4.5: Performance of our framework on iCub images. The system was tested on 108 objects per class, with each object having three views (front, left and right).

Parameter	Cube	Sphere
Precision	0.98	0.95
Recall	0.81	0.74

different views of an object is shown in Figure 4.7.



(a) (b) (c)

Figure 4.7: Different views of an object: (a) front (b) left (c) right.

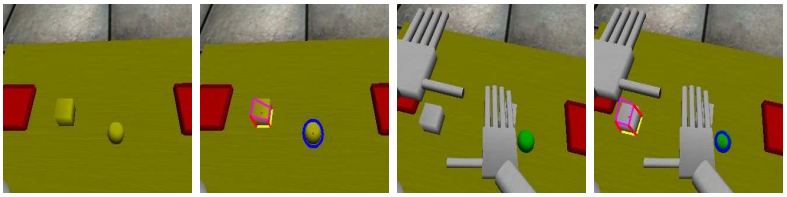
The images were taken from the simulator's camera by moving the robot head systematically across the table, in fixed steps. The dataset is generated with the robot's motor system disabled and the object detection is done offline. The precision and recall for this dataset is reported in Table 4.5.

The experiments depict the utility of our framework which combines probabilistic reasoning with active vision to give fairly accurate results. It helps in reducing the amount of visual evidence to be processed by selective tuning to relevant regions of the image, and also helps in reasoning about the object category in an incremental and modularized way. For example, to detect cubes, we check for L-structures in the initial run of inference and fine tune the processing to regions where L's have been detected. Complex operators such as corner points are then applied on these regions and are grouped to form subsequent higher level structures such as squares and cubes.

The sphere class has less precision than cube since it is inferred from only one object part *sector* in a single layer (*layer 1*), while cubes are inferred from multiple object parts across four layers. As the inference progresses up the hierarchy, the category becomes

Table 4.6: Comparison of detection accuracy for independent views and objects.

Accuracy	Cube	Sphere
detection-per-view	80.55%	74.38%
detection-per-object	98.14%	92.59%



(a) (b) (c) (d)

Figure 4.8: Detection on iCub images: (a)-(b) without motor system (c)-(d) with motor system. The different sub-parts and their intersections are highlighted: pink square-face, yellow L-face and red intersections.

more defined reducing false positives. This also explains the less recall for sphere. If the sectors extracted are not well defined, then they may not be inferred as spheres, since there are no other supportive object parts.

Shifted Views for Active Vision: We also conducted experiments using shifted views of an object as an active vision mechanism, Table 4.6 reports the per class detection accuracy on the above mentioned dataset. The *detection-per-view* depicts the accuracy obtained when each of the three views of an object are considered as independent images. It is compared with *detection-per-object* where an object is considered to be detected if any of its views get detected. As observed from the results, we can gain significant improvement in accuracy if we leverage the different views of an object for detection. The embodied vision system can exploit this advantage since the agent can interact with its environment.

Few examples of inference on 3D objects are shown in Figure 4.8. The figures depict the robot's view of the table at a given instant.

As can be seen from the results, our system is fairly resilient to noise. This is mainly due to the two stage visual processing allowed by our framework. In the initial run of in-

ference, we gather visual evidence at a fairly high threshold for the visual operators, which rules out most of the noisy edges and sectors. Depending on object parts detected from the initial run, active vision with a relaxed threshold is performed on regions around the object parts. This is advantageous over doing a single stage visual processing with the relaxed thresholds, which will create a lot of spurious detections. The use of active vision also provides significant gain in time and memory over a single stage visual processing. The input evidence obtained from iCub images are mostly inexact and inaccurate. In the presence of broken edges or faint lines, our system can perform fairly well since the predicates which check for spatial features such as *intersects*, *isPerpendicular*, etc., are tolerant to such errors to a certain extent. We are able to detect multiple categories of objects across the image in parallel using MLNs as opposed to MRFs which would require replicating the MRF for each category across the image. The objects in different orientations can be detected using a single template, because the predicates are orientation invariant.

4.5 Real Images

In order to validate our system on real images, we also tested on a dataset consisting of images of a Rubik's cube, taken from Microsoft's Kinect Xbox (R). The dataset consisted of 55 images of cubes with different colored faces facing the camera. Each such orientation had 10-11 views (snapshots) taken by rotating the Kinect in fixed steps of five degrees.

4.5.1 Texture Operator

For this experiment, we used texture as the primary operator in the active vision stage. Initially we tested the system with the original set of visual operators reported in Table 4.1. But this set of operators did not work well across all the images in the dataset. This is due to uncontrolled lighting conditions and varying colors of the faces within a single object. This led to the requirement of very low vision thresholds to obtain suitable evidence across the images, which resulted in generation of a large amount of visual evidence. The texture

Table 4.7: Visual routines used at each layer for cube detection using texture operator. The shape properties detected at each layer are indicated in bold.

Layer	Spatial and Shape Predicates
L1	isPerpendicular(line, line), intersectsOrdered(line, line), isDiff(line, line), positiveIntersection(line, line), suitableLength(line, line), angleGtThanFive(line, line), sufficientGap(corner, corner), **lstruct(line,line)**, **cornerpair(corner, corner)**
L2	isDiffCpairs(cpair, cpair), isPerpendicularCpairs(cpair, cpair), suitablelengthCpairs(cpair, cpair), intersectsOrderedCpairs(cpair, cpair), **cpairlstructure(cpair, cpair)**
L3	isTexture(texture), formsCube(texture,cpair-lstruct), **cube(texture, cpair-lstruct)**

operator was introduced to overcome this difficulty. This operator is suitable when the objects are characterized more by their texture than intensity, and the basic edge and corner detection methods cannot be used effectively. The operator is based on the gray-level co-occurence matrix (Haralick *et al.*, 1973) where statistical correlation between image pixels are determined to extract features such as contrast, homogeneity, cluster tendency, entropy, etc.

The detection using texture operator is as follows: in the initial run, we detect L-structures from lines, which are indicators of where to perform active vision processing. The texture and corner points are extracted in the active vision stage. The texture evidence is extracted over contours which cross an area threshold. The corner points are grouped together to form the end points of L-structures similar to the grouping of lines. The object category is then determined based on the texture value and on whether the L-structures formed from corner points overlap with the contour containing the texture. The grouping of corners into L's as well as the texture and corners into a cube is done by the hierarchical MLN. With texture, we do not look for intersections between squares and L's and hence obtain a bounding box over the object as the output. The set of visual routines used at each layer of the MLN for this method is given in Table 4.7.

4.5.2 Results

Few examples of detection on real images are shown in Figure 4.9. The two sets of images show the results with the original set of operators (i.e., without texture) and with texture as the primary visual operator, respectively. We report the precision and recall obtained by this method on the Kinect dataset in Table 4.8.

(a) (b) (c) (d)

Figure 4.9: Detection on images obtained from Microsoft's Kinect Xbox (R) using (a)-(b) corner points (c)-(d) texture, as the primary operator for active vision. (d) shows the bounding box of the object detected using texture.

Experiments using texture operator on other datasets: Similar experiments were conducted on synthetic and iCub image datasets, using texture as the primary operator along with shapes. The precision and recall obtained on the datasets are also reported in Table 4.8. In our experiments, we used texture operator as a replacement for one of the two shape operators needed to infer a higher level structure. For example, instead of looking for two L's to form a square, we check for one L-structure and a suitable value of texture to conclude a square. In this way, we could detect objects from inexact and incomplete data without generating a large amount of evidence. The synthetic images were treated with 1% salt-and-pepper noise. *Detection-per-view* was used for evaluating iCub images and the class considered is cube alone since texture values were not well distinguished for cubes and spheres of the iCub dataset.

The results of experiments with shape-only operators were reported in the previous sections. On comparing those results with Table 4.8 it is observed that the recall improves when texture is used along with shapes as features than when shape operators are used

alone. This is because shape operators are more difficult to extract than texture. Generally, the precision is observed to reduce when texture is used. This is because shape operators are more discriminative among classes than texture and hence produce less false positives. In our experiments, we replace one of the shape operators of a higher-level structure by texture operator. This causes precision to reduce.

4.5.3 Comparison with Baseline Detector

We compare the performance of the proposed hierarchical MLN system with a baseline consisting of only the texture operator. The baseline detector uses a single level MLN to detect cubes based on the texture value and the contour dimensions. The texture evidence is obtained in the same way as described previously, except that it is now extracted in the initial stage instead of lines and sectors. Thus the baseline detector does not run the active vision stage. The comparison of performance of both methods on all the three datasets is reported in Table 4.8. As seen from the results, the precision is less for the baseline detector which uses texture as the only visual feature, whereas the hierarchical MLN is able to combine multiple features in a systematic way, leading to accurate predictions. The recall also reduces for the baseline detector. This is because the proposed system, which uses active vision, is able to perform focused visual processing to generate more accurate evidence than processing on the entire image.

4.6 Summary and Conclusions

In this chapter, we described the experimental evaluation of the proposed framework. The parameters to be considered and the datasets chosen were discussed. The performance of our system was reported for three different datasets: synthetic, iCub and real images. The system was evaluated on incomplete and noisy evidence, a visually guided task for an embodied agent (iCub) and on real images. The experiments with texture operator as a feature were also presented towards the end of the chapter. Comparison of the proposed

Table 4.8: Comparison of performance of the proposed system with a baseline detector comprising of a single level MLN using texture alone as the feature. The evaluation is on all the three datasets.

Method	Precision	Recall
Hierarchical (synthetic, shape-only)	1	0.42
Hierarchical (synthetic, texture and shape)	0.71	0.51
Baseline (synthetic, texture-only)	0.43	0.52
Hierarchical (iCub, shape-only)	0.98	0.81
Hierarchical (iCub, texture and shape)	0.93	0.87
Baseline (iCub, texture-only)	1	0.19
Hierarchical (real, texture and shape)	0.85	0.62
Baseline (real, texture-only)	0.59	0.24

hierarchical system with a baseline single level MLN was discussed.

The following conclusions can be made empirically from the experiments conducted:

- The proposed system is able to perform fairly well with incomplete and noisy evidence, as shown in the experiments on synthetic images.
- The hierarchical MLN can systematically build and infer objects of complex structures. The detection is currently limited to objects with geometric regularities, but real world objects can be detected by suitably modifying the features and MLN rules.
- The experiments with iCub images demonstrate the suitability of the framework as an embodied vision system. The advantages of combining active vision with hierarchical MLN were brought forth through these experiments. The combination helps in robust, reliable and tractable inference.
- The framework is suitable for detection on real images, even with non-uniform object colors and varying lighting conditions.
- The combination of hierarchical MLN with active vision provides fairly high precision and recall, but an optimized implementation can result in better processing time than the current system.

CHAPTER 5

CONCLUSIONS AND FUTURE DIRECTIONS

In this chapter, we summarize our contributions and outline some potential directions for future research.

5.1 Conclusion

In this thesis we proposed a hierarchical approach to handle uncertainty and incompleteness in visual routines and to systematically build and infer complex structures from basic visual operators. We discussed the concept of visual routines which enable visual perception by identifying shape and spatial relations among objects and its sub-parts in an image. The routines, composed of basic visual operators could be applied on the image to form successive incremental representations which allow for focused visual processing and for systematic development of complex vision systems. Following this approach, we proposed a hierarchical architecture for object categorization. The objects were defined as being composed of sub-parts and the spatial relations among the sub-parts. Each layer in the hierarchy inferred the sub-parts associated with it, which were grouped into higher-level structures at subsequent layers. While probabilistic inference could be performed using traditional graphical models which can handle uncertainty and non-i.i.d. data in a principled way, relational models offer a single template to handle generalizations of an object category. We employed Markov logic networks, a statistical relational model for object inference. We also presented a mechanism for targeting additional processing to relevant regions of the input when the evidence is absent or incomplete. This was done using active vision techniques which could process parts of the image at a higher granularity to yield the missing evidence. We evaluated our system on three different datasets and established empirically that we are able to detect fairly complicated objects. As far as we

are aware, this is the first work to have employed MLNs as a hierarchical model for any application.

The hierarchical modeling of objects helps in systematic inference of objects. It also helps in pruning of object categories as the inference progresses up the hierarchy, since only those objects with sufficient supportive evidence from lower layers need to be inferred at a particular layer. The active vision techniques facilitates tractable inference by filtering irrelevant data which would otherwise have to be processed and would cause inefficiency in processing time and memory. It also provides reliable and more accurate prediction as the agent can improve its initial guess of the object category by obtaining finer details of the image. While the experimental results with our approach are quite promising, the current processing time is not as efficient. This could be improved by an optimized implementation of the system.

5.2 Future Directions

Potential future directions from our work include the following:

- Our framework could be employed for other applications such as visually guided navigation of a robot. This could be done by integrating our framework with a reinforcement learning (RL) algorithm. The current perception of the scene would determine the state of the RL agent and influence the next action to be taken.
- The hierarchical MLN model could be applied in other domains like natural language processing where structures of the text could be built and parsed systematically using the model and the ambiguity associated with the words could be modeled through uncertain reasoning.
- A rigorous formalization of the interaction between the different layers could be done and could employ a feedback mechanism that can alter the beliefs at lower-levels based on the evidence at higher levels. Currently, our approach allows the beliefs of object parts at higher layers to be influenced by the evidence passed from lower layers. Active vision also changes the beliefs of object parts but only at the bottom layer, which then indirectly influences the higher layers as the inference progresses up the hierarchy. In order to develop a rigorous formalization, methods should be developed to pass information in a top-down manner without breaking the hierarchy.

- We learned the weights for the MLN clauses. A further extension would be to allow for learning the structure of the clauses themselves.
- Finally, it would be an interesting direction to try and integrate our object detection system with the motor systems of a real robot for visual object manipulations. This would require a real-time optimized implementation of the current system.

APPENDIX A

LIST OF MLN RULES

Complete list of MLN rules used for the experiments described in this thesis:

Synthetic Image Dataset:

Classes: Square, Triangle, Cone, Cylinder, Train

Table A.1: Rules used with synthetic image dataset.

Layer	Weight	Formula
L1	1.02881	isLine(l_1)∧ isLine(l_2)∧ isPerpendicular(l_1, l_2)∧ intersectsOrdered(l_1, l_2)∧ suitableSize(l_1, l_2) ⇒ l-structure(l_1, l_2)
	8.64097	l-structure(l_1, l_2) ⇒ isLine(l_1)∧ isLine(l_2)∧ isPerpendicular(l_1, l_2)∧ intersectsOrdered(l_1, l_2) ∧ suitableSize(l_1, l_2)
	1.25939	isLine(l_1)∧ isLine(l_2)∧ isDiff(l_1, l_2)∧ intersectsOrdered(l_1, l_2)∧ positiveIntersection(l_1, l_2) ∧ angleGtThanFive(l_1, l_2) ⇒ triangletwosides(l_1, l_2)
	8.95668	triangletwosides(l_1, l_2) ⇒ isLine(l_1)∧ isLine(l_2)∧ isDiff(l_1, l_2) ∧ intersectsOrdered(l_1, l_2)∧ positiveIntersection(l_1, l_2) ∧ angleGtThanFive(l_1, l_2)
	2.17363	isLine(l_1)∧ isLine(l_2)∧ isDiff(l_1, l_2)∧isParallelLines(l_1, l_2)∧similarLength(l_1, l_2) ∧suitableDistBwLines(l_1, l_2)⇒ cyl-lines(l_1, l_2)
	11.1749	cyl-lines(l_1, l_2)⇒isLine(l_1)∧ isLine(l_2)∧ isDiff(l_1, l_2)∧isParallelLines(l_1, l_2) ∧similarLength(l_1, l_2)∧suitableDistBwLines(l_1, l_2)
	1.52325	isSector(s_1)∧isSector(s_2)∧isParallelSectors(s_1, s_2)∧isDifferentSectors(s_1, s_2) suitableAsCylinderSectors(s_1, s_2)∧suitableDistBwSectors(s_1, s_2) ⇒ cyl-sectors(s_1, s_2)
	8.56806	cyl-sectors(s_1, s_2) ⇒ isSector(s_1)∧isSector(s_2)∧isParallelSectors(s_1, s_2) isDifferentSectors(s_1, s_2)∧suitableAsCylinderSectors(s_1, s_2)∧suitableDistBwSectors(s_1, s_2)
L2	1.31707	lstruct(ls_1)∧ lstruct(ls_2)∧ intersects(ls_1, ls_2) ⇒ square(ls_1, ls_2)
	4.79728	square(ls_1, ls_2) ⇒ lstruct(ls_1)∧ lstruct(ls_2)∧ intersects(ls_1, ls_2)
	1.05806	triangletwosides(ts)∧isLine(l)∧closedThreeSides(ts,l)∧closedObj(ts,l) ⇒ triangle(ts,l)
	4.55805	triangle(ts,l) ⇒ triangletwosides(ts)∧isLine(l)∧closedThreeSides(ts,l)∧closedObj(ts,l)
	0.627196	triangletwosides(ts)∧isSector(s)∧closedObj(ts,s)∧fullCurve(s) ⇒ cone(ts,s)
	6.19496	cone(ts,s) ⇒ triangletwosides(ts)∧isSector(s)∧closedObj(ts,s)∧fullCurve(s)
	0.598444	cylinderLines(cyl)∧cylinderSectors(cys)∧formsCylinder(cyl,cys) ⇒ cylinder(cyl,cys)
	5.02635	cylinder(cyl,cys) ⇒ cylinderLines(cyl)∧cylinderSectors(cys)∧formsCylinder(cyl,cys)
L3	1.10231	cubeface(f)∧isSquare(f)∧cylinderSectors(cys)∧abutsWheel(f,cys)⇒trainBody(f,cys)
	4.96002	trainBody(f,cys) ⇒ cubeface(f)∧isSquare(f)∧cylinderSectors(cys)∧abutsWheel(f,cys)
L4	0.0415759	cubeface(f)∧isSquare(f)∧tbody(tb)∧abutsFront(tb,f)⇒trainBodyFront(tb,f)
	5.23914	trainbodyFront(tb,f) ⇒ cubeface(f)∧isSquare(f)∧tbody(tb)∧abutsFront(tb,f)
L5	0.466811	cubeface(f)∧isSquare(f)∧tbfront(tbf)∧abutsTop(tbf,f)⇒train(tbf,f)
	5.81873	train(tbf,f) ⇒ cubeface(f)∧isSquare(f)∧tbfront(tbf)∧abutsTop(tbf,f)

iCub Dataset:

Classes: Cube, Sphere

Table A.2: Rules used with iCub image dataset.

Layer	Weight	Formula
	0.500284	isLine(l_1)∧ isLine(l_2)∧ isPerpendicular(l_1, l_2)∧ intersectsOrdered(l_1, l_2)∧ suitableLength(l_1, l_2) ⇒ l-structure(l_1, l_2)
	8.66214	l-structure(l_1, l_2) ⇒ isLine(l_1)∧ isLine(l_2)∧ isPerpendicular(l_1, l_2)∧ intersectsOrdered(l_1, l_2) ∧ suitableLength(l_1, l_2)
L1	0.500001	isCorner(c_1)∧isCorner(c_2)∧sufficientGap(c_1, c_2) ⇒ cornerpair(c_1, c_2)
	4.1772	cornerpair(c_1, c_2) ⇒ isCorner(c_1)∧isCorner(c_2)∧sufficientGap(c_1, c_2)
	0.470015	isSector(s)∧fullCurve(s)∧suitableSize(s)∧isRound(s)⇒sphere(s)
	6.3587	sphere(s)⇒isSector(s)∧fullCurve(s)∧suitableSize(s)∧isRound(s)
	0.500213	cornerpair(cp_1)∧cornerpair(cp_2)∧isDiffCpairs(cp_1, cp_2)∧isPerpendicularCpairs($cp_1.cp_2$) ∧suitablelengthCpairs(cp_1, cp_2)∧intersectsOrderedCpairs(cp_1, cp_2) ⇒ cpairlstructure(cp_1, cp_2)
L2	8.06527	cpairlstructure(cp_1, cp_2)⇒ cornerpair(cp_1)∧cornerpair(cp_2)∧isDiffCpairs(cp_1, cp_2) ∧isPerpendicularCpairs($cp_1.cp_2$) ∧suitablelengthCpairs(cp_1, cp_2)∧intersectsOrderedCpairs(cp_1, cp_2)
	4.1772	cpairlstructure($cpls_1$)∧cpairlstructure(cp_2)∧cpairL-intersects($cpls_1, cpls_2$)⇒squarecorners($cpls_1, cpls_2$)
L3	0.500001	squarecorners($cpls_1, cpls_2$)⇒cpairlstructure($cpls_1$)∧cpairlstructure(cp_2)∧cpairL-intersects($cpls_1, cpls_2$)
L4	0.607206	squarecorners(cpsq)∧lstruct(ls)∧isTranslated(cpsq,ls)⇒cube(cpsq,ls)
	5.03081	cube(cpsq,ls)⇒squarecorners(cpsq)∧lstruct(ls)∧isTranslated(cpsq,ls)

Real Image Dataset:

Class: Cube

Table A.3: Rules used with real image dataset.

Layer	Weight	Formula
	0.500284	isLine(l_1)$\wedge$ isLine(l_2)$\wedge$ isPerpendicular(l_1, l_2)$\wedge$ intersectsOrdered(l_1, l_2)$\wedge$ suitableLength(l_1, l_2) $\Rightarrow$ l-structure(l_1, l_2)
	8.66214	l-structure(l_1, l_2) $\Rightarrow$ isLine(l_1)$\wedge$ isLine(l_2)$\wedge$ isPerpendicular(l_1, l_2)$\wedge$ intersectsOrdered(l_1, l_2) $\wedge$ suitableLength(l_1, l_2)
L1	0.500001	isCorner(c_1)$\wedge$isCorner(c_2)$\wedge$sufficientGap(c_1, c_2) $\Rightarrow$ cornerpair(c_1, c_2)
	4.1772	cornerpair(c_1, c_2) $\Rightarrow$ isCorner(c_1)$\wedge$isCorner(c_2)$\wedge$sufficientGap(c_1, c_2)
	0.500213	cornerpair(cp_1)$\wedge$cornerpair(cp_2)$\wedge$isDiffCpairs(cp_1, cp_2)$\wedge$isPerpendicularCpairs($cp_1.cp_2$) $\wedge$suitablelengthCpairs(cp_1, cp_2)$\wedge$intersectsOrderedCpairs(cp_1, cp_2) $\Rightarrow$ cpairlstructure(cp_1, cp_2)
L2	8.06527	cpairlstructure(cp_1, cp_2)$\Rightarrow$ cornerpair(cp_1)$\wedge$cornerpair(cp_2)$\wedge$isDiffCpairs(cp_1, cp_2) $\wedge$isPerpendicularCpairs($cp_1.cp_2$) $\wedge$suitablelengthCpairs(cp_1, cp_2)$\wedge$intersectsOrderedCpairs(cp_1, cp_2)
	0.7	cpairlstructure(cpls)$\wedge$isTexture(tx)$\wedge$formsCube(cpls,tx)$\Rightarrow$textureCube(cpls,tx)
L3	3.81423	textureCube(cpls,tx))$\Rightarrow$cpairlstructure(cpls)$\wedge$isTexture(tx)$\wedge$formsCube(cpls,tx)

Publications

Communicated:

1. **Priya Anna Mani**, **Sriraam Natarajan**, and **Balaraman Ravindran** (2012), A Hierarchical Markov Logic Based Framework for Reasoning with Incomplete Visual Evidence. Communicated to Knowledge and Information Systems - An International Journal, Springer.

REFERENCES

Agre, P. and **D. Chapman**, Pengi: An implementation of a theory of activity. *In Proceedings of the Sixth National Conference on Artificial Intelligence*. Seattle, WA, 1987.

Antanas, L., **M. Otterlo**, **M. Oramas**, **T. Tuytelaars**, and **L. Raedt**, Not far away from home: A relational distance-based approach to understand images of houses. *In Proceedings of the 20th International Conference on Inductive Logic Programming*. Florence, Italy, 2010.

Awasthi, P., **A. Gagrani**, and **B. Ravindran**, Image modeling using tree structured conditional random fields. *In Proceedings of the 20th International Joint Conference on Artificial Intelligence*. Hyderabad, India, 2007.

Bala, J., **K. DeJong**, **J. Huang**, **H. Vafaie**, , and **H. Wechsler**, Visual routine for eye detection using hybrid genetic architectures. *In Proceedings of the 13th International Conference on Pattern Recognition*. Vienna, Austria, 1996.

Barenholtz, E. and **J. Feldman**, Interpretation of part boundaries and the movement of attention. *In 1st Annual Meeting of the Vision Sciences Society*. Sarasota, Florida, 2001.

Baylis, G. C. and **J. Driver** (1994). Parallel computation of symmetry but not repetition in single visual objects. *Visual Cognition*, **1**, 377–400.

Biederman, I. (1987). Recognition by components: A theory of human image understanding. *Psychological Review*, **94**, 115–147.

Bradski, G. (2000). The opencv library. *Dr. Dobb's Journal of Software Tools*.

Bruynooghe, M., **D. Fierens**, **B. Gutmann**, **A. Kimmig**, **N. Landwehr**, **W. Meert**, **I. Thon**, and **L. D. Raedt**, An exercise with statistical relational learning systems. *In Proceedings of the International Workshop on Statistical Relational Learning*. Leuven, Belgium, 2009.

Chachoua, M. and **D. Pacholczyk** (2002). Qualitative reasoning under ignorance and information-relevant extraction. *Knowledge and Information Systems*, **4:4**, 483–506.

Chapman, D. (1991). *Vision, Instruction and Action*. Ph.D. thesis, MIT.

Chechetka, A., **D. Dash**, and **M. Philipose**, Relational learning for collective classification of entities in images. *In Proceedings of Association for the Advancement of Artificial Intelligence Workshop on Statistical Relational AI*. Atlanta, USA, 2010.

Cooper, P. R. and **P. N. Prokopowicz**, Markov random fields can bridge levels of abstraction. *In Proceedings of Neural Information Processing Systems*. Denver, CO, 1991.

Domingos, P. and **D. Lowd**, *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan & Claypool, San Rafael, CA, 2009.

Dubba, K. S. R., **A. G. Cohn**, and **D. C. Hogg**, Event model learning from complex videos using ilp. *In Proceedings of the 19th European Conference on Artificial Intelligence*. Lisbon, Portugal, 2010.

Felzenszwalb, P. F., **R. B. Girschick**, **D. McAllester**, and **D. Ramanan** (2010). Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32:9**, 1627 – 1645.

Fierens, D., **H. Blockeel**, **M. Bruynooghe**, and **J. Ramon**, Logical bayesian networks and their relation to other probabilistic logical models. *In Proceedings of the 15th International Conference on Inductive Logic Programming*. Bonn, Germany, 2005.

Getoor, L. and **B. Taskar**, *Introduction to Statistical Relational Learning*. MIT Press, 2007.

Gilaie-Dotan, S., **S. Ullman**, **T. Kushnir**, and **R. Malach** (2001). Shape-selective stereo processing in human object-related visual areas. *Human Brain Mapping*, **15**, 67–79.

Haralick, R. M., **K. Shanmugam**, and **I. Dinstein** (1973). Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, **SMC-3:6**, 610–621.

Henderson, J. M. (2003). Human gaze-control during real-world scene perception. *TRENDS in Cognitive Sciences*, **7:11**, 498–504.

Horswill, I., Visual routines and visual search: a real-time implementation and an automata-theoretic analysis. *In Proceedings of the 15th International Joint Conference on Artificial Intelligence*. Montreal, Canada, 1995.

Ijsselmuiden, J. and **R. Stiefelhagen**, Towards high-level human activity recognition through computer vision and temporal logic. *In Proceedings of the 33rd Annual German Conference on Artificial Intelligence*. Karlsruhe, Germany, 2010.

Jaeger, M., Parameter learning for relational bayesian networks. *In Proceedings of the Twenty-Fourth International Conference on Machine Learning*. Oregon, USA, 2007.

Jaeger, M. (2008). Model-theoretic expressivity analysis. *Probabilistic Inductive Logic Programming, Lecture Notes in Computer Science*, **4911**, 325–339.

Johnson, M. P. (1993). *Evolving Visual Routines*. Master's thesis, MIT.

Kersting, K. and **L. D. Raedt**, *Bayesian Logic Programming: Theory and Tool, In Introduction to Statistical Relational Learning*, chapter 10. MIT Press, 2007.

Kok, S., **M. Sumner**, **M. Richardson**, **P. Singla**, **H. Poon**, **D. Lowd**, and **P. Domingos** (2007). The Alchemy system for statistical relational AI. Technical report, University of Washington, Seattle, WA.

Kumar, S. and **M. Herbert**, Discriminative fields for modeling spatial dependencies in natural images. *In Proceedings of the Advances in Neural Information Processing Systems 16*. Vancouver, Canada, 2003.

Lafferty, J., **A. McCallum**, and **F. Pereira**, Conditional random fields: Probabilistic models for segmenting and labeling sequential data. *In Proceedings of the 18th International Conference on Machine Learning*. Williamstown, MA, 2001.

Lowd, D. and **P. Domingos**, Efficient weight learning for markov logic networks. *In Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*. Warsaw, Poland, 2007.

Mailis, T., **G. Stoilos**, and **G. Stamou** (2010). Expressive reasoning with horn rules and fuzzy description logics. *Knowledge And Information Systems*, **25:1**, 105–136.

Marr, D. (1976). Early processing of visual information. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, **275:942**, 483–519.

Metta, G., **G. Sandini**, **D. Vernon**, **L. Natale**, and **F. Nori**, The iCub humanoid robot: an open platform for research in embodied cognition. *In Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems, ACM*. Gaithersburg, Maryland, 2008.

Nunez-Varela, J. I., **B. Ravindran**, and **J. L. Wyatt**, Where do I look now? Gaze allocation during visually guided manipulation. *In Proceedings of the IEEE International Conference on Robotics and Automation (to appear)*. Saint Paul, MN, 2012.

Poon, H. and **P. Domingos**, Sound and efficient inference with probabilistic and deterministic dependencies. *In Proceedings of the 21st National Conference on Artificial Intelligence*. Boston, MA, 2006.

Qin, B., **Y. Xia**, and **S. Prabhakar** (2011). Rule induction for uncertain data. *Knowledge And Information Systems*, **29:1**, 103–130.

Raedt, L. D., **A. Kimmig**, and **H. Toivonen**, Problog: A probabilistic prolog and its application in link discovery. *In Proceedings of the 20th International Joint Conference on Artificial Intelligence*. Hyderabad, India, 2007.

Rao, S. (1998). *Visual Routines and Attention*. Ph.D. thesis, MIT.

Sato, T. and **Y. Kameya** (2001). Parameter learning of logic programs for symbolic-statistical modeling. *Journal of Artificial Intelligence Research*, **15**, 391–454.

Shanahan, M. (2005). Perception as abduction: Turning sensor data into meaningful representation. *Cognitive Science*, **29**, 103–134.

Shet, V., **M. Singh**, **C. Bahlmann**, **V. Ramesh**, **J. Neumann**, and **L. Davis** (2011). Predicate logic based image grammars for complex pattern recognition. *International Journal of Computer Vision*, **93**, 141–161.

Singh, M. and **D. D. Hoffman**, Part-based representations of visual shape and implications for visual cognition. *In* **T. Shipley** and **P. Kellman** (eds.), *From Fragments to Objects: Segmentation and Grouping in Vision*, volume 130. Elsevier Science, 2001, 401–459.

Singh, M. and **B. Scholl**, Using attentional cueing to explore part structure. *In Annual Symposium of Object Perception and Memory*. New Orleans, Louisiana, 2000.

Singla, P. and **P. Domingos**, Markov logic in infinite domains. *In Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*. Vancouver, Canada, 2007.

Song-Chun, Z. and **D. Mumford** (2005). Predicate logic based image grammars for complex pattern recognition. *Journal of Foundations and Trends in Computer Graphics and Vision*, **2**, 259–362.

Sridhar, M., **A. G. Cohn**, and **D. C. Hogg**, Learning functional object-categories from a relational spatio-temporal representation. *In Proceedings of the 18th European Conference on Artificial Intelligence*. Patras, Greece, 2008.

Swain, M. J. and **M. A. Stricker** (1993). Promising directions in active vision. *International Journal of Computer Vision*, **11:2**, 109–126.

Tikhanoff, V., **P. Fitzpatrick**, **F. Nori**, **L. Natale**, **G. Metta**, and **A. Cangelosi**, The iCub humanoid robot simulator. *In Proceedings of Intelligent Robots and Systems Workshop on Robot Simulators*. Nice, France, 2008.

Tran, S. D. and **L. S. Davis**, Visual event modeling and recognition using markov logic networks. *In Proceedings of the 10th European Conference on Computer Vision*. Marseille, France, 2008.

Ullman, S. (1984). Visual routines. *Cognition*, **18**, 97–156.

Warden, T. and **U. Visser** (2011). Real-time spatio-temporal analysis of dynamic scenes. *Knowledge And Information Systems*, **27**, 1–37.

Weng, C. and **Y. Chen** (2010). Mining fuzzy association rules from uncertain data. *Knowledge And Information Systems*, **23:2**, 129–152.

Wu, C. and **H. K. Aghajan**, Recognizing objects in smart homes based on human interaction. *In* **J. Blanc-Talon** (ed.), *Advanced Concepts for Intelligent Vision Systems, Part II, Lecture Notes in Computer Science*, volume 6475. Springer-Verlag, Berlin, Heidelberg, 2010, 131–142.