ABSTRACT

Title of dissertation: BILATTICE BASED LOGICAL REASONING FOR AUTOMATED VISUAL SURVEILLANCE AND OTHER APPLICATIONS

Vinay D. Shet
Doctor of Philosophy, 2007

Dissertation directed by: Professor Larry Davis
Department of Computer Science

The primary objective of an automated visual surveillance system is to observe and understand human behavior and report unusual or potentially dangerous activities/events in a timely manner. Automatically understanding human behavior from visual input, however, is a challenging task. The research presented in this thesis focuses on designing a reasoning framework that can combine, in a principled manner, high level contextual information with low level image processing primitives to interpret visual information. The primary motivation for this work has been to design a reasoning framework that draws heavily upon human like reasoning and reasons explicitly about visual as well as non-visual information to solve classification problems. Humans are adept at performing inference under uncertainty by combining evidence from multiple, noisy and often contradictory sources. This thesis describes a logical reasoning approach in which logical rules encode high level knowledge about the world and logical facts serve as input to the system from real world observations. The reasoning framework supports encoding of multiple rules for the same proposition, representing multiple lines of reasoning and also supports encoding of rules that infer explicit negation and thereby potentially contradictory information. Uncertainties are associated with both the logical rules that guide reasoning as well as with the input facts. This framework has been applied to visual surveillance problems such as human activity recognition, identity maintenance, and human detection. Finally, we have also applied it to the problem of collaborative filtering to predict movie ratings by explicitly reasoning about users preferences.

BILATTICE BASED LOGICAL REASONING FOR AUTOMATED VISUAL SURVEILLANCE AND OTHER APPLICATIONS

by

Vinay Damodar Shet

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2007

Advisory Committee:
Professor Larry Davis, Chair/Advisor
Professor V. S. Subrahmanian
Professor Rama Chellappa
Professor Ramani Duraiswami
Professor Min Wu

Acknowledgements

I would like to thank a number of people who provided constant support and encouragement throughout my graduate studies and to who I owe the successful completion of my degree. First and foremost I would like to thank my advisor, Professor Larry Davis for providing me with invaluable guidance, the opportunity to work on challenging and interesting problems, and the latitude to pursue my own research ideas. A special thanks is due to David Harwood for helping me start working on high level reasoning for visual surveillance. I would also like to thank Professor V. S. Subrahmanian, Professor Lise Getoor and Professor Jeff Horty for the lengthy discussions we have had on matters related to logical reasoning and for helping me understand nuances of different reasoning paradigms. I would also like to thank Professor Ramani Duraiswami, Professor Rama Chellappa and Professor Min Wu for agreeing to serve on my dissertation committee, for their comments and advice.

A number of individuals at Siemens Corporate Research, were instrumental in seeing a some key ideas to fruition. Discussions with Dr. Ramesh Visvanathan, helped raise important questions about advantages/disadvantages of the bilattice based logical reasoning approach which in turn, spurred me on extend the system further and apply it to more challenging problems. I am also thankful to Dr. Jan Neumann, my collaboration with whom helped realize these extensions into working models and strong papers.

My colleagues, friends and roommates have made graduate school a fun place to be and deserve special thanks, V. Shiv Naga Prasad, Ser-Nam Lim, Sameer Shird-

honkar, Vikas Raykar, Son Tran, Aniruddha Kembhavi, Vlad Morariu, Alap Karapurkar, Indrajit Bhattacharya, Prithviraj Sen, Vasu Parameshwaran, Jan Ernst, Vijay Gopalakrishnan, Christopher Kommareddy, Arunchandar Vasan, Sadagopan Srinivasan, Ishan Banerjee and many more. Many of these people also helped by acting in my staged videos for different surveillance scenarios that are part of this thesis and I am deeply indebted for that.

Finally, I would like to thank my family - my parents Damodar and Varija Shet and my wife Vidya for being patient and standing by my side during the ups and downs of graduate school. I dedicate this thesis to you.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The primary objective of an automated visual surveillance system is to ensure safety and security of the environment within which it is deployed by observing and understanding human behavior and reporting unusual or potentially dangerous activities or events in a timely manner. Automatically understanding human behavior from visual input, however, is a challenging task. Successfully accomplishing this task requires the system to (a) take visual input from possibly multiple cameras, (b) identify objects of interest (c) classify these objects into known types (d) track the objects while they are within the field of regard of the cameras, (e) log the occurrence of basic events such as object interactions, and finally (f) employ these basic events to reason about occurrence of various activities of interest, possibly spanning large intervals of time.

This task, however, is made challenging by the ubiquitous presence of uncertainty within all components of this pipeline. Detecting objects and classifying them into various classes depends heavily on the initial segmentation given to us by methods such as background subtraction. Object occlusions and multiple object interactions not only further complicate these tasks but also adversely affect the system's capacity to track and detect the occurrence of basic events. Finally, recognition of human behavior based on the occurrence of these basic events can fail in the face of exceptions to the definition of that behavior. Since these uncertainties are inherent in the problem we are trying to solve, they cannot be trivially eliminated. Therefore, the system must employ some mechanism that will allow it to deal not only with uncertainty in the definition of these activities/events, but also uncertainties associated with the occurrence of each of the basic events. The system therefore needs to have the capacity to reason in the presence of uncertainties. It is also important to note that such a system should respect the real-time demands of a surveillance system, reporting events of interest as and when they occur.

1.1 Overview

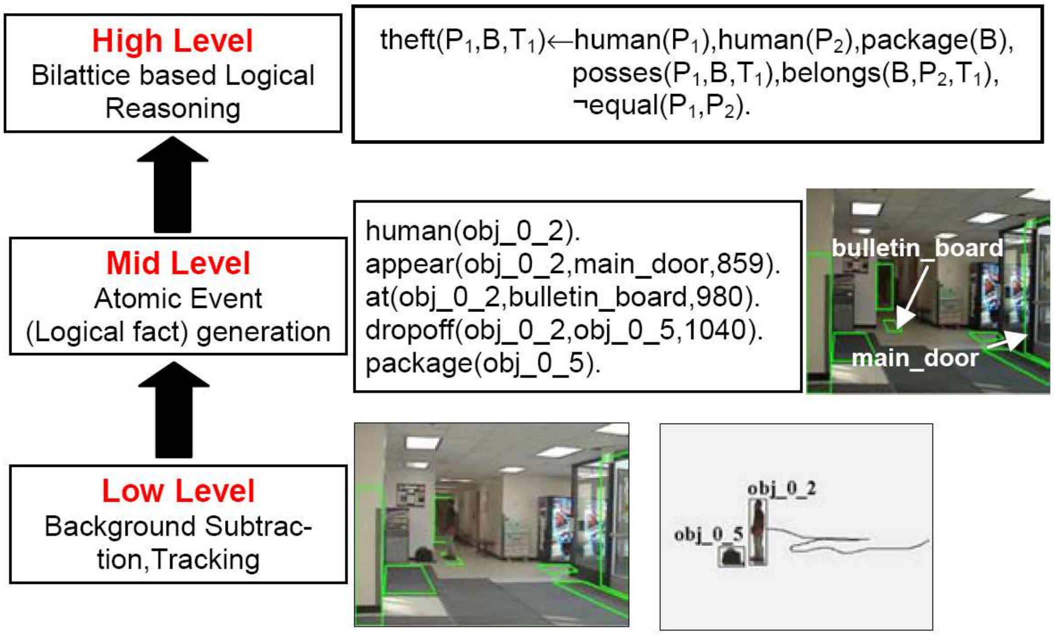
We model the automatic visual surveillance system as a passive, rational agent capable of defeasible reasoning under uncertainty. This agent perceives the occurrence of various events in the video, represents these events as logical formulae, incorporates these formulae in its knowledge base and finally infers their consequences. Among others, these consequences typically include activities of interest like safety violations or security breaches that have been encoded within the agent by the system programmer. Two important modules of this artificial agent are the perception module and the reasoning module.

Any agent functioning in a real world must be capable of gathering new information by sensing its environment. The agent's perception module takes visual input from one or more cameras, generates events of interest from it, represents them appropriately and incorporates them in its knowledge base. The perception module employs a suite of computer vision algorithms, ranging from background subtraction [55] and tracking to appearance based human identification, to achieve this. Unfortunately, the process of perception is seldom exact and it induces a number of uncertainties in the system that the reasoning module needs to overcome.

The reasoning module permits the agent to arrive at logically valid conclusions given input from the perception module. The agent maintains facts and rules in its knowledge base. Rules are typically encoded by the system programmer and facts are assimilated from the perception module. Since information gathered about its environment is uncertain, it is imperative that the agent be equipped with robust mechanisms to not only reason with potentially erroneous information, but also to recover from mistakes it might commit.

The primary motivation for our research has been to design a high level reasoning framework that draws heavily upon human like reasoning and reasons explicitly about visual as well as non-visual information to solve classification problems. Humans are very adept at performing inference under uncertainty by combining evidence from multiple, noisy and sometimes contradictory sources. We use a logical reasoning approach in which logical rules encode high level knowledge about the world and logical facts serve as the input to the system from real world observations. The reasoning framework encodes multiple rules for the same proposition representing different lines of reasoning and also supports encoding of rules that infer explicit negation and thereby potentially contradictory information.

Figure 1.1 shows the overall reasoning framework for a visual surveillance application. The low level module takes video/image input and performs initial


High Level
Bilattice based Logical Reasoning
Mid Level
Atomic Event (Logical fact) generation
Low Level
Background Subtraction,Tracking
bulletin_board
main_door
obj_0_2
obj_0_5

$$\begin{aligned}\text{theft}(P_1,B,T_1) \leftarrow\ & \text{human}(P_1),\text{human}(P_2),\text{package}(B),\\ & \text{posses}(P_1,B,T_1),\text{belongs}(B,P_2,T_1),\\ & \neg\text{equal}(P_1,P_2).\end{aligned}$$

```
human(obj_0_2).
appear(obj_0_2,main_door,859).
at(obj_0_2,bulletin_board,980).
dropoff(obj_0_2,obj_0_5,1040).
package(obj_0_5).
```

Figure 1.1: Overview of system architecture for logical reasoning based activity recognition for automated visual surveillance.

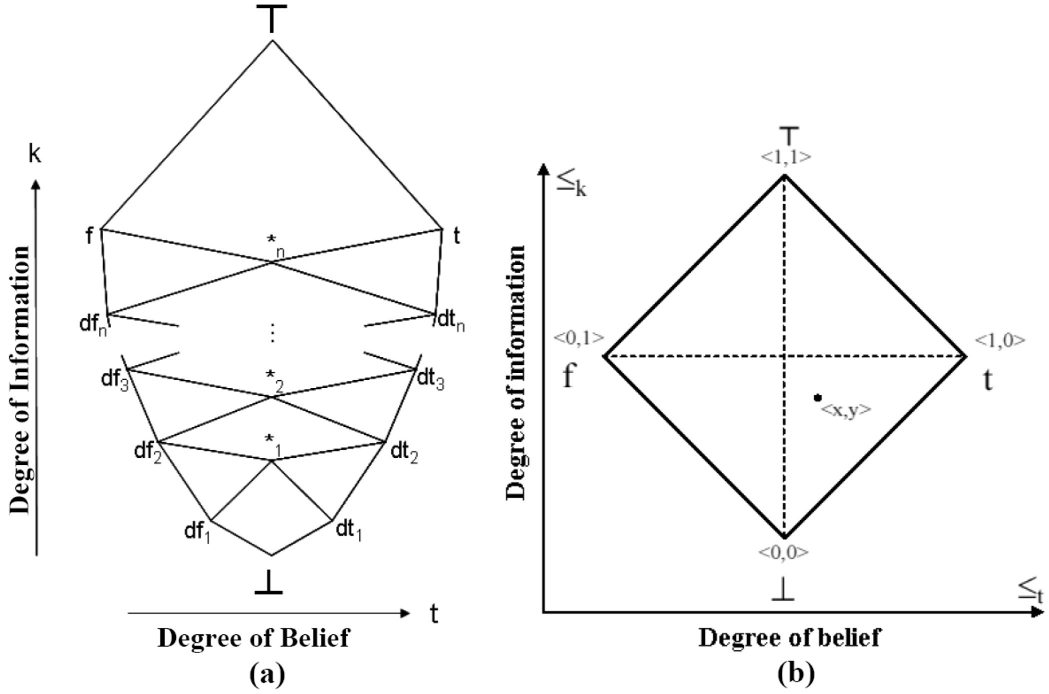


Figure 1.2: Uncertainties assigned to logical rules and facts are taken from a set structured as a Bilattice (a) Bilattice for prioritized default logic (b) Bilattice for continuous valued logic where every element is of the form:$\langle evidence\_for, evidence\_against \rangle$.

processing on it such as background subtraction, tracking, object detection etc. This information is then passed on to the mid level module where atomic patterns of interest are recognized and syntactically structured as logical facts. This module can also take as input pre-specified annotations of the observed scene (such as the "bulletin-board" and "main_door" in Figure 1.1) to assert a semantically richer set of logical facts. These logical facts are then inserted into the knowledge base of the high level reasoning module. This module uses these facts in conjunction with rules encoded in the logic programming language to arrive at valid inferences.

Uncertainties are associated with both the logical rules that guide reasoning (encoding degree of confidence of the rule) as well as the input facts (encoding confidence of observation). These uncertainty values are taken from a set structured as a bilattice. Bilattices are algebraic structures introduced by Ginsberg [6] as a uniform framework within which a number of diverse applications in artificial intelligence can be modeled. These uncertainty measures are ordered along two axes, one along the source's degree of information and the other along the agent's degree of belief. This structure provides a uniform framework which not only permits encoding multiple rules for the same proposition, but also allows inference in the presence of contradictory information from different sources. Figure 1.2(a) shows an example of a discrete valued bilattice where uncertainties associated with the logical rules and facts range from the usual t, and f to $\perp$, $\top$, dt_i, and df_i (i=1 to n) denoting different degrees of uncertainty. The semantics of the reasoning in this case follow that of prioritized default logics. Figure 1.2(b) shows a continuous valued bilattice where every element is of the form evidence_for, evidence_against and is suitable for applications where reasoning in the continuous uncertainty domain is required. The central vertical line represents the line of indifference. If the final uncertainty value computed for a hypothesis lies on this line, it indicates that the agent is indifferent about whether to completely accept or completely reject the hypothesis, either because it has no information or because its sources contradict each other.

The intuition behind using a bilattice representation is that every piece of knowledge, be it a rule or an observation from the real world, provides a different degree of information. An agent that has to reason about the state of the world based on this input will have to translate between the source's degree of information and its own degree of belief. Ideally, the more information a source provides, the more strongly an agent is likely to believe it (i.e. closer to the extremities of the degree-of-belief axis), the only exception to this rule being the case of contradictory information. When two sources contradict each other, it will cause the agent's degree of belief to decrease despite the increase in information content. Such a logic based reasoning approach also generates proofs or justifications for each classification it makes. These justifications (or lack thereof) are further employed by the system to explain and validate, or reject potential decisions. These proofs are also available to the end user as an explanation of why the system has made a particular classification. This framework also allows for top-down control feedback, driven by the high level reasoning.

1.2 Applications

We have applied the framework described above to a number of applications including human activity recognition, identity maintenance and human detection within the domain of visual surveillance. For activity recognition, we employed the reasoning framework to detect various activities of interest in surveillance video, such as thefts, leaving packages behind, unauthorized entries into secure buildings, etc. We also maintained the identities of all humans performing these activities, across short visibility gaps, such as those caused by occlusions, to much longer visibility gaps, resulting from humans, entirely leaving the field of view and later returning.

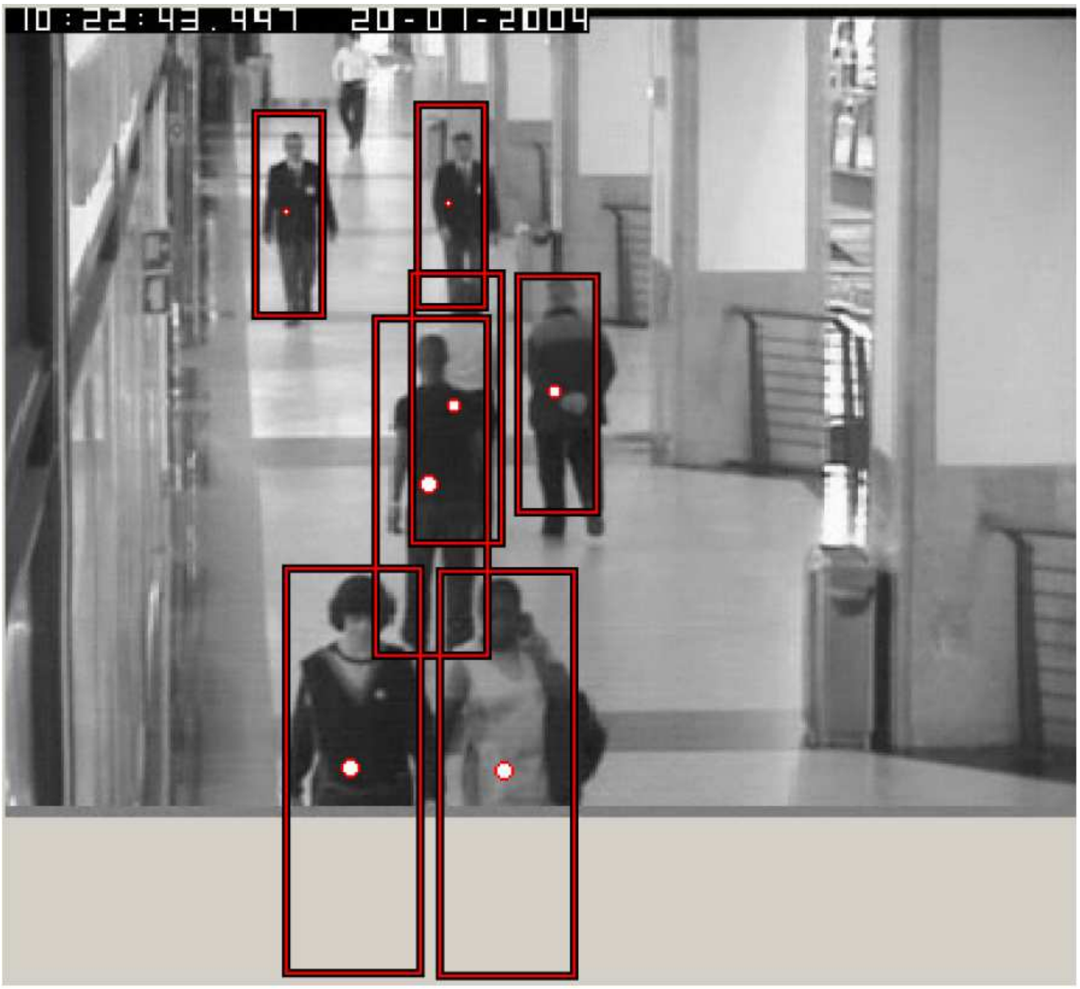


Figure 1.3: Example of human detection using the bilattice based logical reasoning approach.

Identity was maintained in both cases by augmenting various contextual cues in the environment to low level image processing primitives and explicitly reasoning about identity. For the human detection application, the low level module detected various body parts such as heads, torsos and legs using a boosted cascade of gradient histogram based detectors, while the high level reasoning module imposed contextual, scene geometry and human body constraints and also explicitly reasoned about inter-human occlusions. The results obtained by applying this framework to images taken from well known datasets compare favorably with the best previously reported in the literature. Figure 1.3 shows the results of human detection on a single frame from one such dataset.

We also applied this framework to the problem of preference modelling and specifically worked on predicting movie ratings for a given user based on his/her historical preferences as well as preferences of other users. The bilattice based logical reasoning framework was able to naturally model this problem and performed favorably compared with state of the art approaches in the field of preference modelling and ranking. We have also attempted to compare theoretically, the bilattice based reasoning approach to traditional Bayesian reasoning approaches to obtain a better understanding of its strengths and weaknesses.

1.3 Organization of thesis

The thesis is organized as follows: Chapter 2 covers some background material needed for various aspects of this thesis including preliminaries of logic programming, historical development of the theory behind reasoning, previous work on activity recognition, identity maintenance, tracking, occlusion handling, human detection and preference modelling. Chapter 3 describes the application of the logical reasoning framework for activity recognition and lists the different activities that the framework was successfully able to recognize. Chapter 4 motivates the problem of identity maintenance in the context of activity recognition and describes how default logic can be used to address this problem. It then goes on to describe how bilattice based multivalued logics can be used to explicitly reason about identities as well as the activities performed by humans. Chapter 5 raises the issue of handling certain situations in visual surveillance where a strict bottom-up approach to computing is insufficient. It makes the case for the need to have top-down feedback built into a surveillance system and demonstrates how the bilattice based multivalued logic framework can be used to model this top-down feedback in addition to the bottom-up information flow. Chapter 6 addresses one of the fundamental problems in surveillance, that of actually detecting humans in the scene under difficult conditions such as partial occlusions. We describe how we employ a continuous version of the bilattice to do this and also report results and experimentally compare our approach with a state-of-the-art human detection approach. Chapter 7 describes the application of the reasoning framework to the problem of collaborative filtering to predict movie ratings. We describe the methodology and also experimentally compare our results with those of other state-of-the-art preference modelling and ranking approaches. Finally, in chapter 8 we theoretically compare the bilattice based logical reasoning approach with statistical approaches such as Bayesian networks and approximations to full Bayesian inference such as Naive Bayes and Noisy-OR. We conclude in chapter 8.

Chapter 2

Background

In this chapter we will provide background and survey some related work in the areas of reasoning, logic programming, identity maintenance and tracking, activity recognition, human detection, and preference modelling and ranking.

2.1 Reasoning

Systems that take in new information and revise their belief set to maintain consistency are called belief revision systems. These systems are logical frameworks for modelling the dynamics of knowledge. A lot of work, spanning several decades, has been done on belief revision [2, 76, 68, 18, 19] especially in building common sense reasoning systems.

The theoretical framework within which different models of belief revision are embedded is called an epistemological theory. The main task of such a theory is to provide a conceptual apparatus to deal with the problems of knowledge maintenance and change. There are several epistemic factors that constitute the core of an epistemological theory [36]. The most important of them is the *states of belief* or *epistemic states*. This corresponds to the set of beliefs that the agent maintains at a certain point of time. Belief change can be interpreted as moving from one state to another. The second factor is a classification of the agent's *epistemic attitudes*.

This corresponds to the status of various beliefs the agent harbors. For example an agent might accept or reject a certain fact as true or assign some degree of possibility to it. The third factor is an account of the *epistemic inputs*. These are facts that are added to the belief set which might result in belief changes. The final factor is a classification of *epistemic changes*. Different facts when added to the epistemic state might trigger different kinds of epistemic changes.

2.1.1 Theory of Belief Revision

The most popular theory of belief revision so far has been the AGM theory [2, 36] proposed by Carlos Alchourrón, Peter Gärdenfors, and David Makinson. The AGM theory defines three different kinds of belief changes: *expansion, contraction* and *revision*. Given some proposition a and a belief set K, let us assume an agent's epistemic state to be either

- State 1: if it accepts a (i.e. $a \in K$)
- State 2: if it rejects a (i.e. $\neg a \in K$) or
- State 3: if it is indeterminate (i.e $a \notin K$ and $\neg a \notin K$)

Expansion of K by a, K_a^+, then becomes the operation of going from State 3 to either State 1 or State 2. Contraction, K_a^- is going from either State 1 or State 2 to State 3. Revision, K_a^* is going from State 1 to State 2 or vice versa . The AGM theory defines a number of axioms for each of these epistemic changes. Expansion, contraction and revision can be defined as a modification to K such that: $K_a^+ = Cn(K \cup \{a\})$, $K_a^- \nvdash a$ and $K_a^* = (K_{\neg a}^-)_a^+$(Levi's identity) respectively.

When a belief that is logically consistent with the original belief set is added, the agent believes in the logical closure of the original set plus the new belief. On the other hand, addition of a belief that is inconsistent with the existing set, causes the agent to retreat to the most entrenched of the maximal subsets, of the existing set, that are consistent with the new belief, adding the new proposition to that set and closing under logical consequence. In other words, weakly held beliefs are more vulnerable to change in face of a contradiction than strongly held ones.

2.1.1.1 Problems

Although the AGM theory is very popular, it has some drawbacks. In [37], Hansson points out that even though the AGM model of belief change is simple and elegant, it fails to capture several features of real world belief systems because of this simplification. The only way to make it more realistic, according to [37] is to subject it to various amendments and extensions, rendering the theory less mathematically elegant.

One of the most controversial properties [82, 19] of the revision operators is the axiom of success. Success specifies that new information has primacy over existing beliefs of the agent. This property does not seem plausible in many real world applications because in many cases it is not reasonable to give priority to information just because it is new. Another property of the theory which has drawn criticism is that it allows for beliefs to persist long after their justifying beliefs have disappeared.

It must be noted that the AGM model assumes the revision operators to function over deductively closed belief spaces i.e. the belief set is assumed to be closed under consequence, $K = Cn(K)$. It has been noted that this is computationally intractable to support in a real world system [52].

Finally, the most significant limitation of this theory is that it has no representation in the object-language for conditionals. In fact, Gärdenfors himself proved [29] that it is not possible to include any conditional satisfying the Ramsey's test ($a \rightarrow b \iff b \in K_a^*$) without trivializing the revision operator.

2.1.2 Defeasible Reasoning

In first order logic, a conditional of the form $\forall X bird(X) \rightarrow fly(X)$ can be interpreted to mean: if there exists any entity with the property of being a bird, then it has to have the property of being able to fly. There is no way of saying, for example,

$$\begin{aligned} \forall X bird(X) &\rightarrow fly(X) \\ \forall X penguin(X) &\rightarrow \neg fly(X) \\ bird(tweety) & \\ penguin(tweety) & \end{aligned} \tag{2.1}$$

without entailing a contradiction. However, commonsense tells us that as humans, we can accept the rule "birds fly" and "tweety is a bird that cannot fly" without contradicting ourselves.

Reasoning is *defeasible* when the corresponding argument is rationally compelling but not deductively valid [57]. Defeasible conditionals are those in which if the antecedent is true then "normally" the consequent is true. Thus the sentence $\alpha \rightarrowtail \beta$ can be interpreted as "if α holds then β normally holds". Thus using defeasible reasoning it is possible to say $bird(X) \rightarrowtail fly(X)$ meaning "birds normally fly", but there could exist exceptions. Any argument that causes this conditional to fail is called a defeater for that rule. In this case "penguin is a bird that cannot fly" is a defeater for "birds fly".

2.1.2.1 Belief Revision based on Defeasible Reasoning

Rott [76] and Pollock [71] have pointed out that a possible solution of avoiding triviality on inclusion of defeasible conditionals within the AGM framework, is to desist from interpreting the belief set as a closed, single set of beliefs. They argue that, one must make a sharp distinction between foundational or explicit beliefs, denoted by K_e and derived or inferred beliefs, denoted by $K_i (= K - K_e)$.

Belief change can then be modelled on the assumption that new beliefs are added to K_e (which is logically consistent with the existing set of those beliefs). Beliefs added can be inconsistent with previously inferred beliefs belonging to K_i. The new belief set K', consists simply of the closure of the new explicit set, K'_e, under the relation of defeasible consequence, $K' = dCn(K'_e)$.

Note that the operator for closure under defeasible consequence, dCn is non-monotonic, meaning that it does not necessarily obey $A \subset B \Rightarrow dCn(A) \subset dCn(B)$. Also note, in this case defeasible rules are explicitly represented among the agent's beliefs and $dCn(K_e)$ stays contradiction free because these defeasible rules are provisioned to handle exceptions. This view is further bolstered by Gärdenfors and Makinson [26] who argue that belief revision and nonmonotonic reasoning are "two sides of the same coin". It is interesting to note that we are no longer required to have a deductively closed belief set.

2.2 Logic Programming

It is now time to move from a mathematical setup to a more practical programming framework. A logic programming language, like its first order counterpart, consists of constant symbols (i,e. the "individuals" in the world), function symbols (mapping of individuals to individuals) and predicate symbols (mapping from individuals to truth values). Atoms are of the form $p(t_1, \cdots, t_n)$, where the t's are terms and p is a predicate symbol of arity n.

2.2.1 Normal Logic Programs

Definition 1 (Normal Logic Programs $\mathcal{N}$)**.** *A normal logic program*[1] *is a finite set of rules of the form* $A_0 \leftarrow A_1, \cdots A_m,\ not\ A_{m+1}, \cdots,\ not\ A_n$*, where each* A_i *is an atom and 'not' is a logical connective called negation as failure.*

The left hand side of the rule is called the *head* and the right hand side is referred to as the *body*. If the body of the rule is an empty set, the rule is denoted by $A \leftarrow$ or simply A and is referred to as a $fact$. Note that logic programs make

[1] Normal logic programs that do not have *not* are called *definite logic programs*

the closed world assumption, meaning that they assume that the knowledge base contains everything that is required to be known and failure to find an atom means it is false. Also note that *not* is not allowed to occur in the head of a rule.

It is interesting to observe that negation by failure bestows upon logic programs a defeasible nature. For instance we can implement the example from subsection 2.1.2 in the following manner

$$\begin{array}{rcl} fly(X) & \leftarrow & bird(X),\ not(abnormal(X)). \\ abnormal(X) & \leftarrow & penguin(X). \\ & & bird(tweety). \\ & & bird(alfred). \\ & & penguin(tweety). \end{array} \tag{2.2}$$

and conclude $fly(alfred)$ and $not(fly(tweety))$. Note, we would conclude $fly(tweety)$ in the absence of $penguin(tweety)$ so the addition of extra information about *tweety* has caused us to retract our belief that tweety can fly. In this case, $abnormal(tweety)$ is a defeater for $fly(tweety)$. It is also interesting to note that since *not* is not allowed in the head of the rule, we will never encounter a situation where some a and $not(a)$ are simultaneously true. In other words, normal logic programs are by design contradiction free.

2.2.2 Extended Logic Programs

Definition 2 (Extended Logic Programs $\mathcal{E}$). *Normal programs that have been extended with the classical (explicit) negation operator, denoted by* $\neg$*, in addition to* *the (implicit) negation by failure are called extended logic programs.*

The classical negation is needed when it is not possible to sensibly make the closed world assumption. In such cases, absence of information might indicate ignorance which could be resolved at a later time. One also needs classical negation if one wishes to explicitly infer negative information, e.g $\neg fly(X) \leftarrow penguin(X)$. (Note that *not* is not allowed in the head of the rule).

Unfortunately, the introduction of classical negation and its ability to reside in the head of a rule, can cause an extended logic program to be inconsistent. It is therefore imperative that if we wish to follow the belief revision strategy outlined in section 2.1.2.1, we need to make sure that our default rules account for the exceptions. Reverting again to the *tweety* example, we now get:

$$\begin{aligned} fly(X) &\leftarrow bird(X),\ not(\neg fly(X)). \\ \neg fly(X) &\leftarrow penguin(X). \\ & \quad bird(tweety). \\ & \quad bird(alfred). \\ & \quad penguin(tweety). \end{aligned} \tag{2.3}$$

Note that in this case too we get the same results as we did previously but now our rule has a statement of the form $not(\neg fly(X))$. This can be interpreted as an attempt to disprove the negation of what is about to be proven.

2.2.3 Defeasible Extended Logic Programs

A defeasible logic program is defined in terms of two distinct sets of rules. A set of rules representing definite knowledge and a set of defeasible rules representing tentative information.

Definition 3 (Definite Rule Θ)**.** *A definite rule is an ordered pair of the form* $Head \leftarrow Body$ *where* $Head$ *is a literal and* $Body$ *is a set of literals such that the* $Body$ *never contains the connective* 'not'*. Literals could be of the form* a *or* $\neg a$ *(denoting classical negation).*

Definition 4 (Defeasible Rule Δ)**.** *A defeasible rule is an ordered pair, denoted* $Head \multimap Body$ *where* $Head$ *is a literal and* $Body$ *is a set of literals which could be of the form* a *or* $\neg a$ *(denoting classical negation) and where the body may contain the connective* 'not'*.*

Definition 5 (Defeasible Logic Program $\mathcal{P}$)**.** *A defeasible logic program,* $\mathcal{P}$*, is a finite set of definite and defeasible rules denoted by* Θ *and* Δ *respectively. When needed we shall denote* $\mathcal{P}$ *by* (Θ, Δ)*.*

Defeasible derivation for a query Q given $\mathcal{P}$ is obtained by backward chaining from Q using both definite as well as defeasible rules. It is assumed that the set of definite rules, Θ is contradiction free meaning that there are no derivations possible from it alone that prove some a and $\neg a$.

2.3 Identity Maintenance and Tracking

Identity maintenance in surveillance has typically only employed some form of appearance matching. [67] uses a SVM based approach to recognize individuals in indoor images based on color and shape based features. [4] employs gait as a characteristic to identify individuals while [98] performs face recognition from video. Microsoft's *EasyLiving* project [58] employs two stereo cameras to track up to 3 people in a small room while [89] describes a multi-camera indoor people localization in a cluttered environment.

One of the main issues to be handled by any tracking system is that of ocluions. Object occlusions have been handled in literature either explicitly or implicitly. Pfinder [92] is one such such system that tracks objects implicitly. It represents models of humans by a collection of colored blobs, deletion or addition of which during and after occlusions helps it handle partial occlusions. [47] uses closed-world regions to perform context-based tracking of multiple objects with erratic movement and collisions. [38], [64], [61] use region tracks and appearance models to identify people after occlusions. [54] maintains a list of persons and classifies pixels into foreground or background. [48] presents a Bayesian blob-tracker which implicitly handles occlusions by incorporating the number of interacting persons into the observation model and inferring it using Bayesian Network. [50] accounts for occlusions by an outlier component in a generative appearance model and use online EM to learn and update the parameters of this model.

Among systems that explicitly reason about occlusions, [94] incorporates an extra hidden process for occlusion into a dynamic Bayesian network, and relies on the statistical inference of the hidden process to reveal occlusion relations. [78] uses appearance models to localize objects and uses disputed pixels to resolve the object's depth ordering during occlusions. [56] tracks vehicles by using a ground plane constraint to reason about vehicle occlusions. Layer representation has also been used to model occluding objects. [73] represents self-occlusion with layered templates, and uses a kinematic model to predict occlusions. [10] automatically decomposes video sequences into constituent layers sorted by depths by combining spatial information with temporal occlusions. [53] and [85] both model videos as a layered composition of objects and use EM to infer objects appearances and motions. [99] extend [85] by introducing the concept of background occluding layers and explicitly inferring depth ordering of foreground layers.

2.4 Activity Recognition

A significant body of work on activity recognition has employed some form of state-based representations such as Hidden Markov Models and their extensions. Starner and Pentland in [84] use HMMs to recognize hand movements for American Sign Language. More complex models, such as Parameterized-HMMs (PHMM) [91] have been used to model actions with underlying parameters like the direction of a pointing gesture. Coupled-HMMs (CHMM) [6], are designed to model the interaction between two agents by coupling the states of two HMMs while, variable length-HMMs [28] have been used to capture behavioral dependencies and constraints between events.

State-based models can also be used to model higher level behavior if the states are taken to represent higher level meanings. Bobick and Ivanov [49], propose the use of a stochastic context-free grammar to compute the probability of a temporally consistent sequence of primitive actions recognized by HMMs. Clarkson and Pentland model events and scenes from audiovisual information in [13]. Brand and Kettnaker in [5] propose an entropic-HMM approach to minimize the entropy of the data and thus organize the observed video activities into meaningful states. In [42], a probabilistic finite-state automaton is used for recognizing different scenarios, such as monitoring pedestrians or cars on a freeway. More recently Bayesian networks have also been adopted for modelling and recognition of human activities [22, 21, 12, 46].

Non state-based models have also been employed for activity modelling. Rota and Thonnat [75], propose an approach for video sequence interpretation based on declarative models of activities. They define scenarios for *Vandalism, Access forbidden and Holdup* and use a hierarchy of facts ranging from abstract to concrete to recognize these situations. [14] investigates the use of qualitative spatio-temporal representations and abduction in an architecture for Cognitive Vision while [8] employs a context representation scheme for surveillance systems.

[43] considers an activity to be composed of action threads and recognizes activities by propagating constraints and likelihood of event threads in a temporal logic network. [33] uses *chronicles*, a temporal representation scheme for time, events and actions, while [88] uses *scenarios* to declare spatio-temporal knowledge

in vision applications. [32] uses Fuzzy Metric-Temporal Horn Logic (FMTHL) to detect vehicle queues from road traffic scenes.

2.5 Human Detection

Human detection is another important application in the field of automated visual surveillance. The primary question of interest is: is it possible to hypothesize about the presence of humans with certain properties at a particular location in the world and then come up with rules to verify which of those hypothesis are viable and are likely to be humans.

The paper by Zhou and Nevatia [97] poses this problem as a state space search problem. They first define a state space, Θ, comprised of states, θ, of the following form $\theta = \{n, \{M_1, M_2, \cdots, M_n\}\}$. Since the number of humans in the scene at any point in time can vary, the complete state space consists of states of varying dimensionality. In other words, $\Theta = \bigcup_{i=0}^{\infty} \theta_n$ where θ_n is the set of states of dimension n. Θ is essentially an infinite space from which a MAP estimate, $\theta^* = argmax_{\theta \in \Theta} P(\theta|I)$, needs to be computed. Since Θ is such a large space, a sampling algorithm has to be used to converge to a good solution and this sampling has to be driven by the data. The authors use a MCMC approach that is driven by jump-diffusion dynamics with low level features like head detectors to converge to the most likely state.

Another state based approach for detecting objects in general is the work proposed by Hoiem et. al [41]. In this work, they model the states of objects in the real world along with other contextual cues such as viewpoint and scene geometry.

The inference problem is then formulated as estimating the states of these objects given observed local evidence. Since context is also being inferred along with the state of the object itself, this method they claim is better than any approach that tries to only detect objects.

There are several approaches that try to detect humans as an integral whole using different kinds of detectors. Papageorgiou et. al. [70] use SVM detectors, Felzenszwalb [20] uses shape models and Gavrilla [31, 30] uses edge templates to recognize full body patterns. Leibe et.al [59] employ an iterative method combining local and global cues via a probabilistic segmentation. The most popular detector used in such systems is a cascade of detector trained using AdaBoost as proposed by Viola and Jones [86]. Such an approach uses as features several haar wavelets and has been very successfully applied for face detection in [86]. Viola and Jones themselves applied this detector to detect pedestrians in [87] and made an observation that Haar wavelets are insufficient by themselves as features for human detection and they augmented their system with simple motion cues to get better performance. Another feature that is increasing in popularity is the histogram of oriented gradients. It was introduced by Dalal and Triggs [16] and they used a SVM based classifier. This was further extended by Zhu et. al [100] to detect whole humans using cascade of histograms of oriented gradients.

Part based representations have also been used to detect humans. Mohan et.al. [66] divide the human into four different parts and learn SVM detectors using Haar wavelet features. Mikolajczyk et. al. [65] divide the human body into seven parts and for each part a Viola-Jones approach is applied to orientation features.

Wu and Nevatia [93] use edgelet features and learn nested cascade detectors [45] for each of several body parts and detect the whole human using a iterative probabilistic formulation.

The problem with all of the approaches discussed above is that very few [97, 41, 93, 59] of them follow up low level detection with high level reasoning. High level reasoning is required to enforce global constraints to weed out false positives and increase accuracy. Even among the approaches that do use high level reasoning, the models they employ are simplistic and they attempt to estimate the state of the objects in the real world directly from weak observations. Moreover, very few [97, 93, 59] attempt to handle inter-human occlusions. Occlusions are again handled by making iterative hypotheses and checking to see if the hypothesis satisfies observation. However, the results of such systems are limited by the simplicity of the model employed.

2.6 Preference Modelling

Learning rankings was first treated as a classification problem on pairs of objects by Herbrich et al [39] and subsequently used on a web page ranking task by Joachims [51]. Algorithms similar to SVMs were used to learn the ranking function. Burges et al. [11], use a neural network (RankNet) to model the underlying ranking function. Similar to our approach it used a gradient descent technique to optimize a probabilistic cost function–the cross entropy. The neural net is trained on pairs of training examples using a modified backpropagation. Herbrich et al. [40] cast ranking as an ordinal regression problem. The actual ranks are modeled as intervals on the real line. Hence rank boundaries play a critical role during training. The loss function depends on pairs of examples and their target ranks. Several boosting based algorithms have been proposed for ranking. With collaborative filtering as an application Freund et al. [23] proposed the RankBoost algorithm for combining preferences. Dekel et al. [17] present a general framework for label ranking by means of preference graphs and graph decomposition procedure. A log-linear model is learnt using a boosting algorithm. An efficient implementation of the RankBoost algorithm for two class problems was presented in [23]. A convex-hull based relaxation scheme was proposed in [27]. Yan and Hauptmann [95] proposed an approximate margin-based rank learning framework by bounding the pairwise risk function.

Chapter 3

VidMAP:Video Monitoring of Activity with Prolog

3.1 Introduction

Computer Vision based automated surveillance systems are designed to interpret human activity by taking visual input from one or more cameras. In this chapter, we will present a human activity monitoring system called VidMAP. The objectives of the VidMAP system are two fold:

1. To continuously monitor, detect and report predefined violations observed in the input video streams.
2. To answer specific queries about events that have already transpired in the archived video.

Violations are activities that have been "described" to the system by the programmer using a logic programming language. These violations could be transgressions in either security (thefts, unauthorized entry) or safety (unattended packages, collisions). Forensic inquiries into archival footage are typically evoked by a user to check for specific events of interest in the past, like "how many people entered the building between 10:00 am to 1:00 pm?"

A possible violation that the system will have to look for is outlined in the rules below . Assume a surveillance setup where the camera monitors an entrance to a building. Assume also that there exists a card reader, used to control access to the building, that is also observable from the camera's point of view.

Rule 1 (Entry Violation)**.** *An entry violation is defined as the activity of an unprivileged individual entering the building.*

Rule 2 (Privileged Individual)**.** *(a) An individual is privileged to enter a building if she swipes her ID card at the card reader before entering the building.*

(b) An individual is privileged to enter the building without swiping the card, if and only if she is escorted into the building by an individual who is privileged to enter.

(c) Every other individual is unprivileged to enter the building.

Rule 3 (Escort)**.** *An individual A, is considered an escort for individual B, if A is considered a friend of B.*

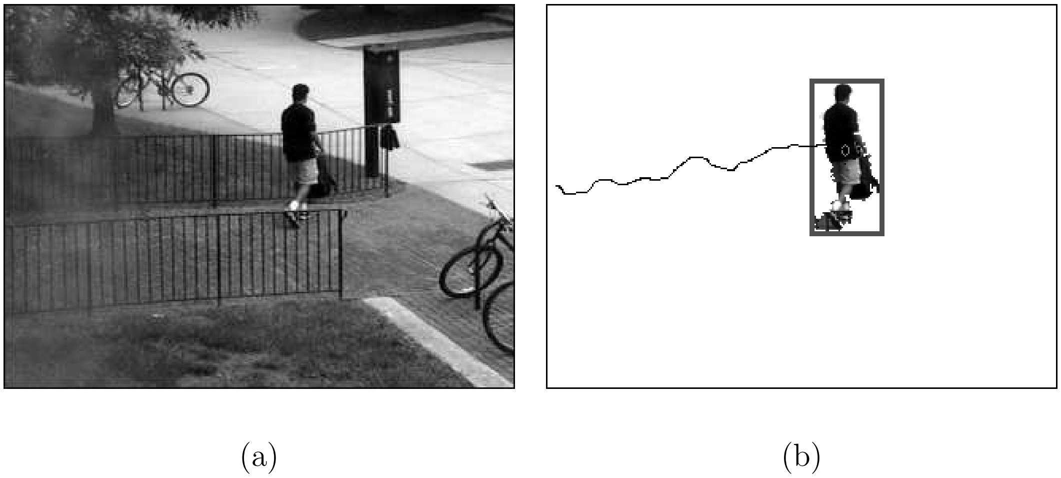
To correctly detect the violation outlined above, any visual surveillance system needs be able to detect primitive events like swiping a card or entering a building. It then needs to establish temporal ordering between these events, specifically in case of this violation, the system needs to check for the absence of the card swipe between the entry of the individual in the scene and her entry into the building. And finally it needs to be able to distinguish between a person being escorted into the building and a person tailgating behind someone else.

We model the automatic visual surveillance system as a passive, rational agent capable of deductive reasoning. This agent perceives the occurrence of various events in the video, represents these events as logical formulae in its knowledge base and infers their consequences. The process of inference is driven by logical rules encoded within the system by the programmer. These logical rules encode various activities of interest which among others, include security and safety violations.

The system is composed of three main modules. A low level image processing module, a mid level fact generation module, and a high level reasoning module.

An agent functioning in a real world must be capable of gathering new information by sensing its environment. The agent's low level module takes visual input from one or more cameras and employs a suite of computer vision algorithms, such as background subtraction [55], tracking and appearance based human identification [96] to provide input to the fact generator. The mid level fact generator recognizes primitive events of interest from this data, and incorporates these events as observed facts in the reasoning module's knowledge base. The high level reasoning module uses these facts in conjunction with rules encoded in the logic programming language, Prolog, to arrive at valid inferences regarding activities observed in the video.

The primary reason for employing a logic programming based approach to recognize activity, over traditional state based approaches, is the expressive power it bestows upon the system that allows us to not only encode complex propositions but also functions and quantification. The use of Prolog differentiates our work from other non-state based approaches as it not only provides us with a ready-to-use mechanism for searching and backward chaining but, as a logic programming language, the semantics and compositionality in Prolog are also well defined.



(a) (b)

Figure 3.1: Figure showing background subtraction and tracking results (a) Single frame from input video stream (b) Background subtracted result with tracking data overlayed

3.2 Low Level Module

The low level computer vision module takes input from one or more cameras and employs a suite of computer vision algorithms to provide input to the higher level modules of the system.

3.2.1 Background Subtraction and Tracking

Surveillance setups typically consist of cameras that are either fixed and observe the same scene at all times or cameras that can perform pan-tilt-zoom operations. In this work, we assume static surveillance cameras and take advantage of the stationary viewpoint by employing a background subtraction algorithm as the first step. We use the code-book based adaptive background subtraction algorithm proposed in [55]. Background subtraction gives us regions that correspond to people, packages, vehicles and other foreground objects of interest.

Tracking of these foreground objects over time permits us to gather temporal properties about them. This in turn helps the fact generator to assert relevant facts about not only the objects themselves, but also their interactions with other objects in the scene. Figure 3.1 shows the results of background subtraction and tracking.

3.2.2 Appearance Matching

Another task the low level module is responsible for is the maintenance of a gallery of previously viewed objects. In any surveillance scenario, it is important to keep track of people as they disappear and reappear from the field of view of the camera. For most part, when people disappear into an open world and later reappear, due to large variability, it is almost impossible to say with certainty that they were observed before or not. However, in a constrained environment, especially one that involves closed worlds like rooms, offices, or even multiple cameras (whose views are adjacent and close to each other) as shown in Figure 3.3, employing an appearance matching technique, in conjunction with pre-established geometric constraints, permits us to reason about people across multiple views.

The algorithm we use for appearance matching is the color path length based approach proposed by [96]. This approach combines color and a geodesic path length measure within the person's body and builds their probabilistic models, which are compared using the Kullback distance. Since spatial information is used for matching the pixels, it is possible to isolate local areas of change within the person.

This enables the system to state whether in one of the views the person was carrying some package. Color information is encoded as either color proportions or ranks of brightness or color. Rank ordering of color information provides for robust matching, especially across multiple cameras.

3.3 Mid Level Module

The main task entrusted to the mid level fact generation module is that of interpreting the data from the image processing module and populating the agent's knowledge base with Prolog facts. These facts correspond to observed events in the video such as an individual entering or exiting a room, or picking up or placing down a package on the floor.

3.3.1 Fact Generation

Data coming in from the low level image processing algorithms can be very noisy. Due to myriad reasons background subtraction routinely introduces artifacts in its output that can get tracked and erroneously labelled as objects of interest. Filtering out such noisy data is of paramount importance for this module. It does so by observing whether or not the object has been persistently tracked. For example, it decides that a tracked object is a human if it satisfies three conditions: it is persistent, it is tall and it has periodic movements associated with it. It considers an object to be a package if it is persistent, does not move on its own and was at some point in time attached to a human. Facts such as entering or exiting a room are generated based on the appearance or disappearance of humans from regions in the image, labelled as *portals*, that are associated with the room.

3.3.2 Background Labelling

For the fact generator to correctly interpret the tracking data, it needs to have an understanding of the scene structure. We provide this information to the fact generator by labelling various areas of the background. The fact generator can now generate facts based not only on object properties and their interactions with each other, but also when objects interact with these labelled background regions. For instance, if we label a region on the ground plane near a vending machine, the fact generator could fire a fact `at(obj_0_14,vending_machine,4439)`, denoting that person labelled `obj_0_14` was observed to be at the vending machine at time 4439, whenever `obj_0_14` appears to pause for some amount of time in that region. Figure 3.4(b) shows a frame at which such a fact is generated by the mid level module.

3.3.3 Sample Facts

Figure 3.2 shows some sample facts that have been generated by this module. We see from this figure that the fact generator recognizes the foreground *blob* labelled `obj_0_2` as a human. It also observes that this human `appears` in the field of view from the region labelled `elevator` at time 859 (in frames). The human appears to pause in a region labelled as `bulletin_board`. The human then `dropsoff` an object labelled `obj_0_5` which is recognized as a `package` and exits the field of view from a region labelled `main_door` at time 1135

```
human(obj_0_2).

appear(obj_0_2,elevator,859).

at(obj_0_2,bulletin_board,980).

dropoff(obj_0_2,obj_0_5,1040).

package(obj_0_5).

disappear(obj_0_2,main_door,1135).
```

Figure 3.2: Sample Prolog facts asserted by the Fact Generator

3.4 High Level Module

The primary task of the high level reasoning module is to continually check to see if the rules that have been encoded in the system can be satisfied by the observed facts that are being asserted into its knowledge base. It does so by backward chaining from each violation or activity and attempts to prove them. In this system, we write rules in and employ a Prolog based inference engine.

The main advantage of employing a logic programming language for specifying activities lies in its expressive nature. If one can describe an activity in plain english, then it can usually be encoded as a logical rule[1]. In this section, we will list en-

[1] It is interesting to note that historically, a primary source of failure for expert systems has been the inability of human experts to translate their expertise into a natural language. Therefore

glish descriptions of a few important violations/activities and provide corresponding Prolog rules that have been encoded in the system.

3.4.1 Thefts

Workplaces are increasingly becoming a target of thefts. Thefts are usually a result of careless employees leaving their belongings inadequately protected. It is therefore important for the surveillance system to recognize the activity of a theft. We define a theft in the following manner:

Rule 4 (Theft)**.** *Theft is the activity of a person possessing an object that does not belong to her.*

Rule 5 (Possess)**.** *A person possesses an object if she carries it.*

Rule 6 (Belong)**.** *An object belongs to an individual if she was seen possessing it before anyone else.*

Rule 4 abstractly captures the essence of stealing while Rule 6 arises out of the assumption that “possession is 9/10ths of the law”. These rules can be encoded in Prolog as follows[2]:

if one were to build an expert system to recognize certain objects, say airplanes, it may never be possible to capture in natural language the processes that humans employ. In our system however, we restrict ourselves to activities that can be “described” to the system in much the same way as a human would to another.

[2] In Prolog, ‘,’ represents a conjunction, ‘not’ represents negation by failure, ‘_’ represents “don’t care” and ‘assert(X)’ inserts Prolog fact X into the knowledge base.

```
theft(P,B,T):- human(P),

               package(B),

               possess(P,B,T),

               not(belongs(B,P,T)).
```

The rule above can be interpreted as P commits a theft of object B at time T if P possess B at time T and at that time, one cannot prove that B belongs to P.

```
possess(P,B,T) :-

          carries(P,B,T).

belongs(B,P,T) :-

          possess(P,B,T),

          not((already_belongs(B,P1,_),

               not(equal(P,P1)))),

          assert(already_belongs(B,P,T)).
```

In the rule for belong above, we insert the predicate already_belongs in the knowledge base to indicate that a person P was first observed possessing the object. Subsequent inquiries into the "belong" status of that object with fail for individuals other than the original P.

3.4.2 Entry Violation

Another important activity that the surveillance system should be aware of is entry violation. Entry violation as described in Rule 1 in Section 3.1, can be encoded in Prolog as follows:

```
entry_violation(P):-
        human(P),
        appear(P,scene,T1)
        enter(P,building_door,T2),
        not(privileged_to_enter(P,T1,T2)).
```

This rule will flag an activity as an entry violation if a person P appears in the field of view of the camera (an event captured by P `appear`ing in region labelled `scene`) at time T1 and enters the building at time T2 such that she is not privileged to enter the building in the time between T1 and T2. Rules 2 and 3 can be encoded as follows

```
privileged_to_enter(P,T1,T2):-
        at(P,card_reader,T),
        T1<T, T<T2.
privileged_to_enter(P,T1,T2):-
        human(P1),
        not(equal(P,P1)),
        at(P1,card_reader,T),
        T1<T, T<T2,
        enter(P1,building_door,T3),
        T1<T3, T3<T2,
        friend(P,P1).
```

The first rule for privilege listed above states that if person P was at the card reader at a time T occurring between T1 and T2, she is privileged to enter the building. The second rules says that in the time between T1 and T2, if there exists another person P1 such that she scans her card and enters the building and if P1 is thought to be a friend of P, then P has the privilege of entry.

3.4.3 Unattended Package

Another violation that is encoded within the surveillance system is that of unattended package.

Rule 7 (Unattended Package)**.** *A package is said to be unattended if it is dropped off by the owner at some time and neither the owner nor a friend of the owner are present in the vicinity of the package at a later time.*

This rule reflects the fact that people can entrust their belongings to their friends to watch over. The corresponding Prolog rule is as follows:

```
unattended_package(B,T):-
     dropoff(P,B,T1),
     disappear(P,_,T),
     T>T1,
     not((friend(P,P1),
          standing_next_to(P1,B,T))).
```

3.4.4 Friend

A person P is considered a friend (in the loosest sense of the word) of P1 by the system if P and P1 are observed standing close and facing each other for a certain amount of time. The concept of friend or acquaintance is important because according to our assumptions an individual can escort her friend into a (semi-secure) building, an individual can entrust her package to a friend to watch over and finally in case some individual A is thought to have committed some violation, then the friends of A could either be accomplices or witnesses.

3.5 Experimental Results

This section describes the implementation details of our system as well as some surveillance scenarios on which it has been tested. The system has been implemented as a real-time, multi-threaded C++ application capable of handling multi-camera scenarios. A Prolog reasoning engine has been embedded within this C++ application.

3.5.1 Implementation Details

The application consists of two kinds of threads: the (possibly multiple) camera thread(s) which are responsible for the low and mid level modules and a single reasoning thread responsible for the high level reasoning module. For each camera connected to the system, we create a camera thread that takes input video frames from the camera assigned to it and runs the low level image processing algorithms and the mid level fact generation routines on it. The reasoning thread, when first created, initializes the Prolog engine that is embedded within it and inserts in its knowledge base, rules for all the predefined activities. The reasoning thread is subsequently evoked every 5 seconds and every time it runs it not only assimilates facts generated by each of the camera threads and inserts them into the Prolog engine, but it also queries the Prolog engine for all violations it is programmed to look for. If in any given run, the Prolog engine is able to prove as true any of the violations, then the reasoning thread alerts the user to it.

The tool we have built also allows the user to manually click on the image, while setting up the system, to mark and label regions in the scene. As mentioned in section 3.3.2, these regions are used by the fact generator to log interactions of objects with the background.

3.5.2 Scenarios

We demonstrate our system in action on a multi-camera surveillance setup as depicted in Figure 3.3. Camera 1 observes the lobby of building with the elevator door, bulletin board, side door, vending machine and main door lying within its field of view, while camera 2 observes the exterior of the building observing a wall mounted phone, a card reader and the entrance to the building. The door to the building does not open unless one swipes their ID card across the card reader.

We tested our system on a continuous 15 minute video clip that contained several violations including entry violations, thefts and unattended packages in ad-

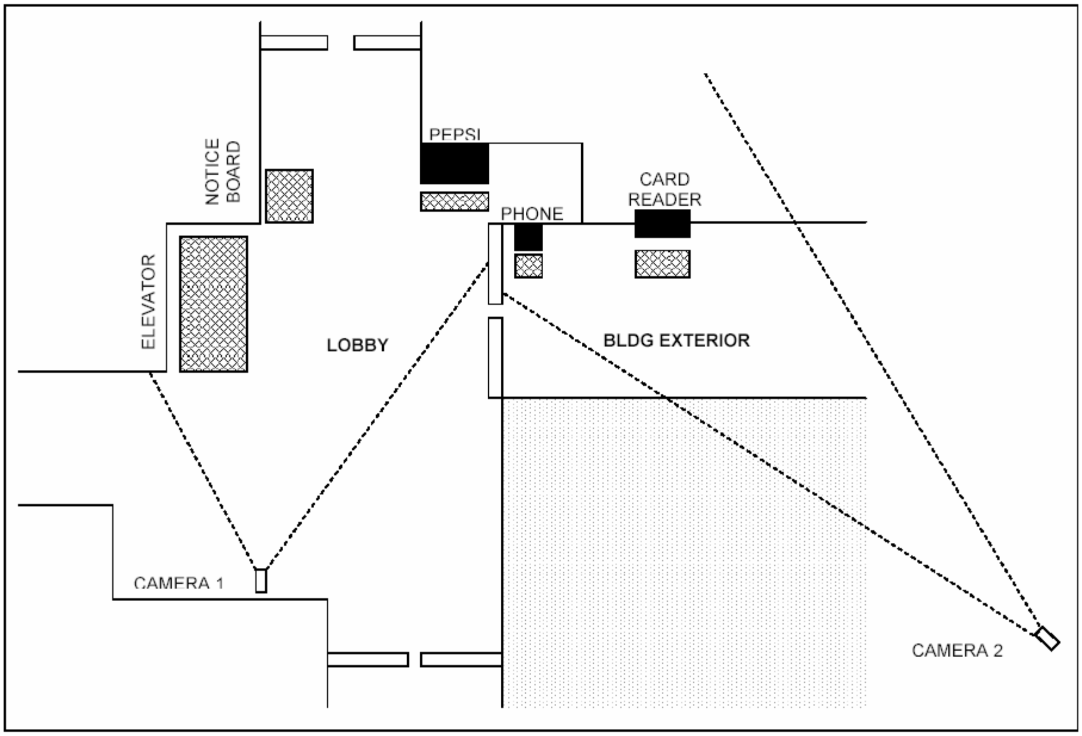


Figure 3.3: Surveillance setup showing fields of view of the cameras in relation to the scene.

dition to significant irrelevant activity. We will now describe each of the scenarios that were used in testing the system[3].

Scenario 1 (Unauthorized Entry)**.** ***(Figure 3.4)*** *Individual 13 in camera 2 swipes his card before entering the building. Individual 15 follows 13 into the building without swiping his card. Once inside, they go different ways.*

Scenario 2 (Escorted Entry, Watching Over)**.** ***(Figure 3.5)*** *Individual 19 in camera 2 swipes his card at the card reader to enter the building while 20 merely follows*

[3] Note: In each of the camera's view, objects are numbered independent of other cameras, as a result a person labelled 15 in camera 2 might appear as 14 in camera 1, however based on appearance and geometric constraints the system is aware of the fact `equal(14_in_1,15_in_2)`.

him in. Once inside, in camera 1 (now referred to as 33 and 34 respectively) they both face each other and start talking. Subsequently, 33 drops off a bag, known to the system as 35, and disappears from view while 34 stands next to it.

In scenario 1, the system cannot prove that 13 is escorting 15 into the building and correctly raises an entry violation, while in scenario 2, initially when 20 follows 19 into the building, the system flags it as a violation but subsequently when the two individuals face and stand next to each other, it assumes they are acquaintances and retracts the entry violation. Subsequently when 33 leaves his bag behind and exits the field of view of camera 1, the system does not raise the unattended package violation as it believes 34 is watching over it.

Scenario 3 (Unattended Package, Witness and Theft)**.** ***(Figure 3.4)*** *Individual 13 drops off package 17, near the bulletin board and exits the field of view of camera 1 through the elevator. Another individual 16, subsequently enters the scene and starts talking with 14. After 16 leaves, 14 picks up package 17 and exits the building.*

When individual 13 exits the scene, since there is no friend of 13 to watch over the package, the system raises an unattended package violation. Subsequently when 16 starts talking to 14, it asserts that 14 and 16 are possible acquaintances. And finally when 14 picks up the bag that originally belonged to 13, the system raises a theft violation and asserts that 16 is a possible witness/accomplice.

In addition to these continuously monitored violations, the application has the ability to take in a custom Prolog query from the user and resolve it using the facts that have been accumulated till that point in time, thus supporting forensic inquiries. For example, to know how many people used the vending machine we can query the system with `at(P,vending_machine,T)` and get back a response `P=obj_0_8, T=2534` and `P=obj_0_14, T=4439` denoting that humans `obj_0_8` and `obj_0_14` were at the vending machine at frames 2534 and 4439 respectively.

3.5.3 Processing Time

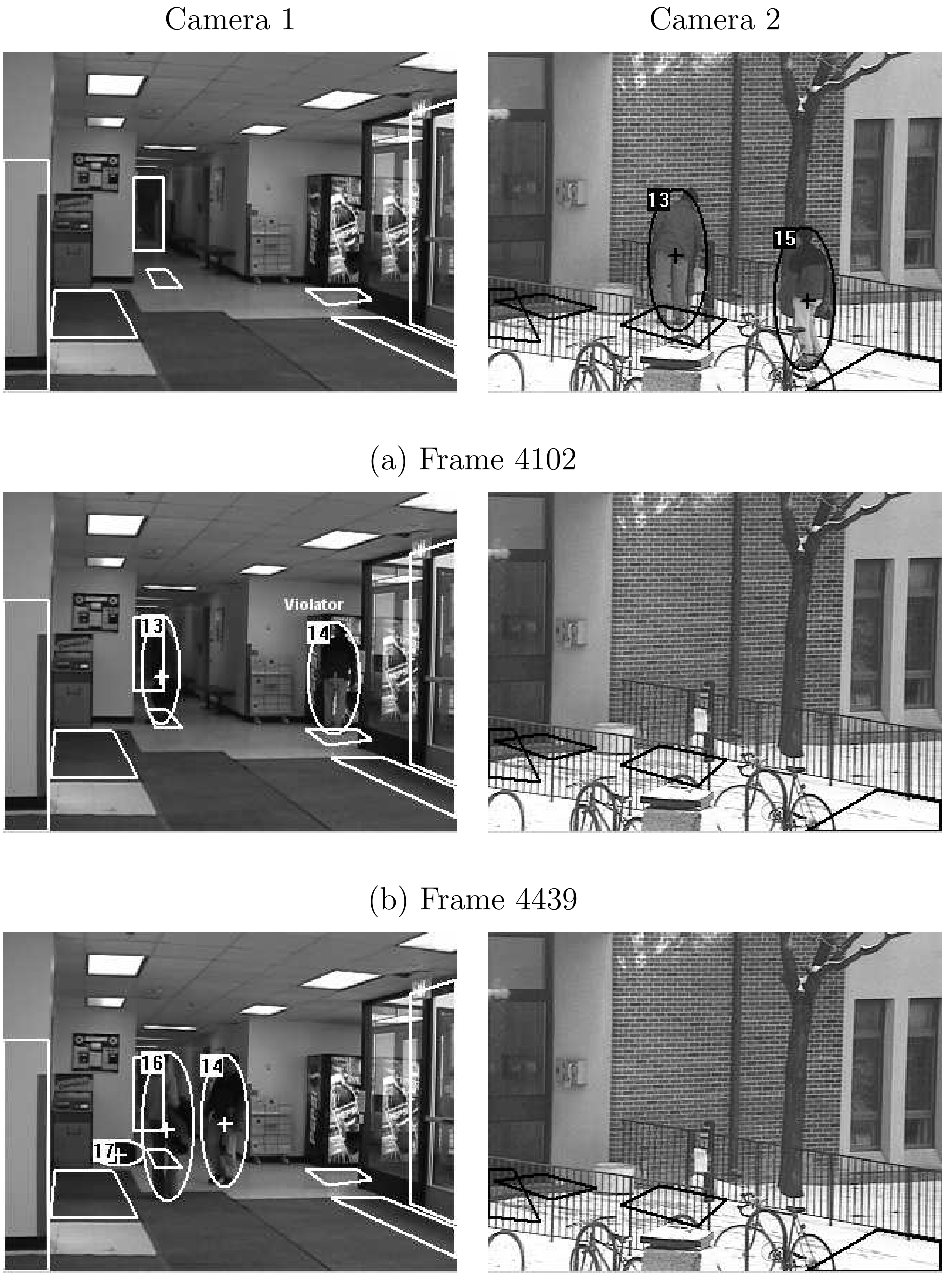
Real time performance is a major concern while building surveillance systems. As the number of facts that the inference engine has to evaluate increases, it is natural to assume that the time taken by it to prove or disprove the occurrence of various activities will increase as well. However, we would like inferences to be made without any significant delay after the event has transpired in the video. To test how the processing time varies as the number of facts in the Prolog knowledge base increase, we ran the original 15 minute video in a loop for several hours and gave that as an input to the system. The original video generated 357 facts. We looped this video 52 times yielding 18564 facts. The time required to infer each of theft, unattended package, entry violation and pickup is shown in Figure 3.6. As can be seen from this graph, beyond around 10000 facts, the inference time varies linearly with the number of facts. These times can be further improved upon by designing the system that forgets past events if they are deemed irrelevant. This will be part of our future work. It is also important to note that due to the multi-threaded architecture of the application, even if the reasoning engine takes time to make inferences, it will not affect the speed of the camera threads which will continue to process the input video streams and generate and assimilate facts from it in real time.

3.6 Discussions and Conclusions

In this chapter, we described the architecture of VidMAP, a visual surveillance system that combines real time computer vision algorithms with Prolog based logic programming, to reason about activities observed in the input video streams. The use of logic programming bestows upon the system expressibility to define various activities. Specifically, using Prolog gives us a ready to use mechanism for searching and backward chaining.

It must be noted that the performance of a system that bases its inferences solely on visual input, depends heavily on the accuracy and scope of its lower level algorithms. Inaccurate output by the low level algorithms can cause the reasoning module to draw incorrect conclusions. For example, if based on appearance, the system erroneously concludes that two humans A and B are not equal, the reasoning engine might conclude that B commits a theft if B picks up a package belonging to A.

The scope of the low level algorithms is important to provide the fact generator with enough information to generate meaningful facts. In the scenarios outlined in the previous sections, we use the theory of a friend. An incorrect theory of a friend can cause the system to erroneously conclude that an illegal entry is legal or that an unattended package is attended. Ideally the low level module should be able to interpret friendly gestures (like shaking hands) between two humans to decide whether or not they are friends. For example, in scenario 2 person 33 could be confronting 34 and reprimanding him for following him into the building, but our system, as of now, will consider them to be friends. However, it must be pointed out that while a human observer has the advantage of being able to interpret subtle body language when people interact with each other, for most part, human reasoning, when constrained to deduction from surveillance data alone, can be erroneous too. Unless there is some explicit body language suggesting that 33 and 34 are not friends, in all likelihood a human observer (who is viewing the surveillance video) may also conclude that no violation has taken place.

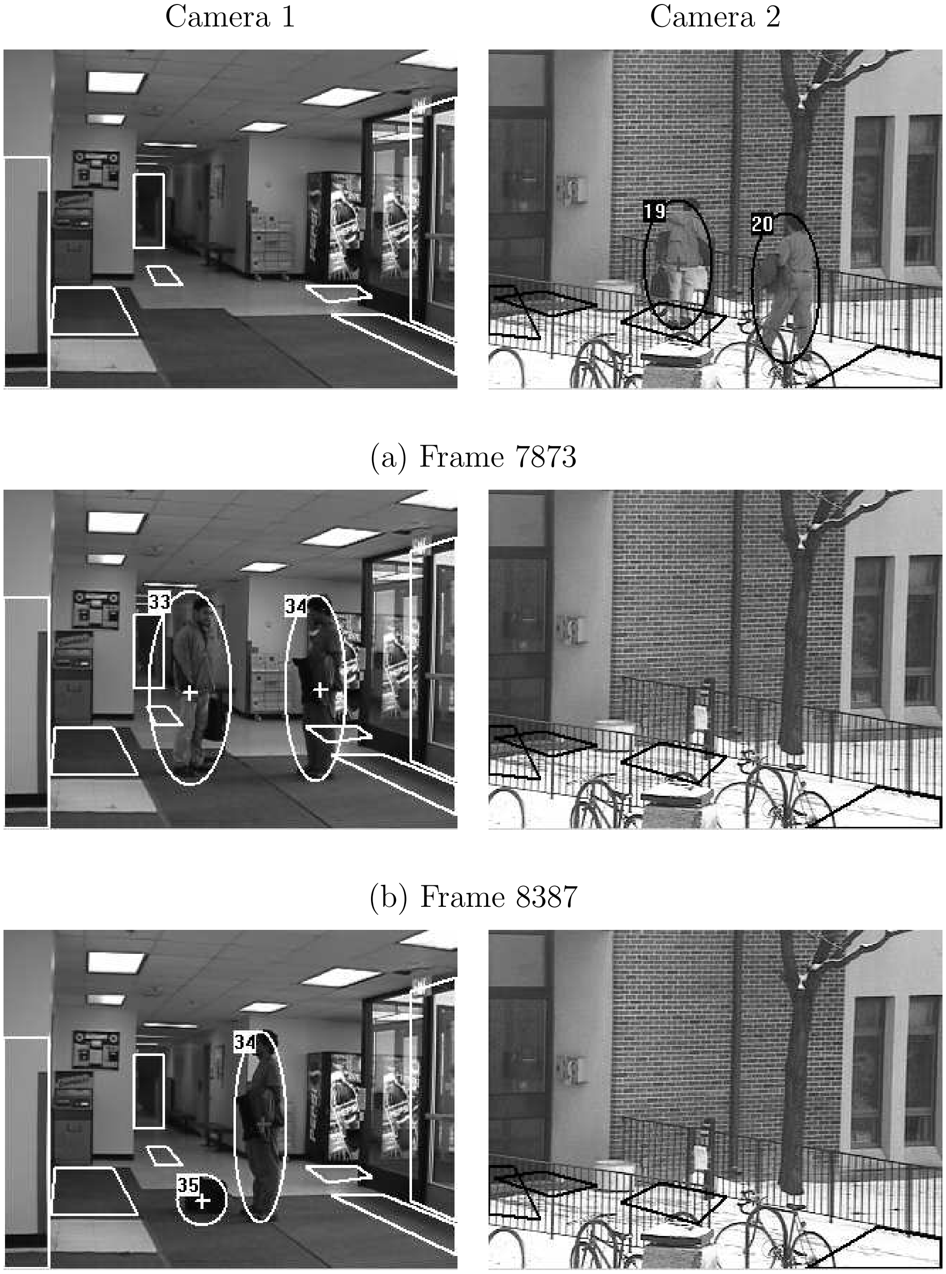


(a) Frame 4102

(b) Frame 4439

(c) Frame 4800

Figure 3.4: (a) Person 15 in camera 2 follows 13 without swiping card. (b) In camera 1 he is referred to as 14. (c) Person 13 in camera 1 drops off bag 17 and disappears from view. Subsequently a different individual, 16 arrives and starts talking with 14.



(a) Frame 7873

(b) Frame 8387

(c) Frame 8523

Figure 3.5: (a)Person 19 in camera 2 swipes his card and enters the building and person 20 follows him in. (b) In camera 1, both individuals now 33 and 34 respectively stand facing each other. (c) Person 34 in camera 1 is standing next to bag 35, left behind by 33.

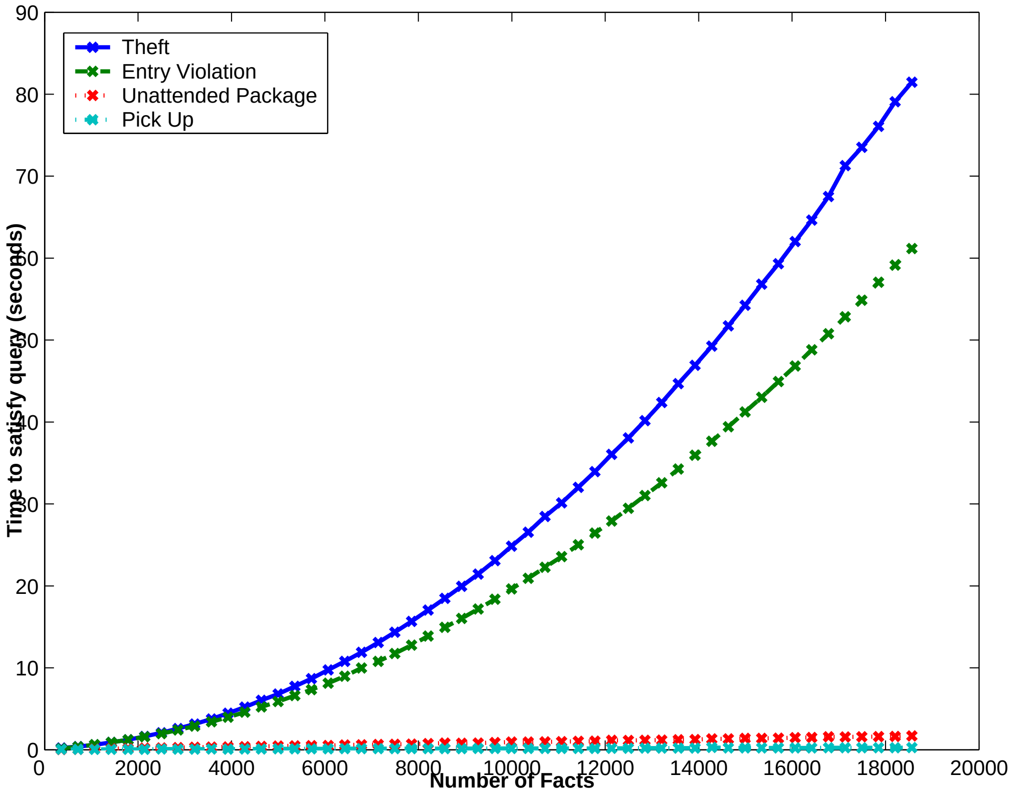


Figure 3.6: Time taken to infer occurrence of certain activities with increasing number of facts.

Chapter 4

Multivalued Logic for Identity Maintenance

4.1 Introduction

The primary goal of a visual surveillance system is to help ensure safety and security by detecting the occurrence of activities of interest within an environment. This typically requires the capacity to robustly track individuals not only when they are within the field of regard of the cameras, but also when they disappear from view and later reappear. Figure 4.1 shows an individual marked X appearing in the scene with a bag, dropping it off in the corridor, and disappearing from view through a door. Subsequently it shows individual Y appearing in the scene through the same door and picking up the bag.

If $individual(X) = individual(Y)$, the activity by itself, is probably not of interest from a security viewpoint. However, if $individual(X) \neq individual(Y)$,

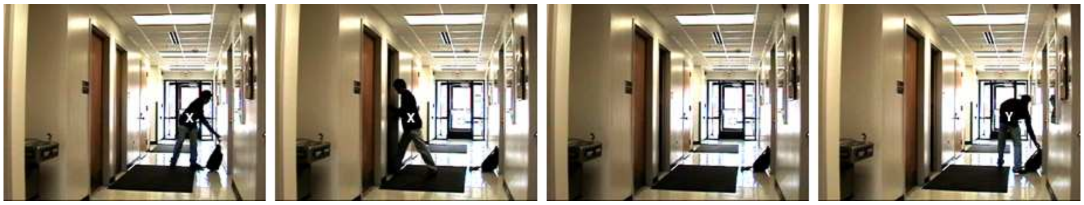


Figure 4.1: Sequence of images showing individual X appearing in the scene with a bag, depositing it on the ground and disappearing from view. Subsequently, individual Y appears in the scene, picks up the bag and leaves.

the activity observed could possibly be a theft. This example captures the general problem of automatically inferring whether two individuals observed in the video are equal or not. This problem is significant not only for camera setups where individuals routinely disappear into and reappear from pockets of the world not observed by the cameras, but also within a single field of view when tracking is lost due to a variety of reasons.

Traditionally in surveillance, the problem of identity maintenance has been addressed by appearance matching. Matching of appearances can be based on a person's color distribution and shape [67], gait [4], face [98] and other physical characteristics. All of these approaches are considered weak biometrics and, by themselves, they are inadequate for maintaining identities for recognizing complex activities.

The objectives of this chapter are to provide a framework

1. **that supports reasoning about identities of individuals observed in video**. We do this by augmenting traditional appearance matching with (a) contextual information about the world and (b) self identifying traits associated with actions. In addition to stating whether or not two individuals are equal, we also qualitatively encode our confidence in it.

2. **that facilitates using this information on identities to recognize activities**. We also propagate our confidence in the identity statements to activities to which they contribute.

In the example above, if the door through which individual X disappeared leads into a closed world (a world with no other exit), we could,under some circumstances, infer that individual Y coming out of that door at a later time had to be equal to individual X (with a high degree of confidence), regardless of whether or not he appeared similar to X.

In this work, we encode contextual information about the world and our common sense knowledge about self-identifying actions as rules in a logic programming language. Furthermore, we observe that since these rules reflect actions taking place in a real world, they can never be definite and completely correct. We therefore employ default logic as the language to specify these rules, which provides our framework the important property of nonmonotonicity (the property of retracting or disbelieving old beliefs upon acquisition of new information). We also employ a bilattice based multivalued representation that encodes our confidence in various rules and propagates these confidence values to the identity statements and subsequently to the activities themselves. We then use prioritization over these default rules to capture the fact that different cues could provide us with different amounts of information. Finally, we use this information about identities of individuals to reason about the occurrence of activities in the video.

4.2 Motivation

Our primary motivation is to build a visual surveillance system that draws heavily upon human reasoning. While humans are very skillful in matching appearances, even we commit mistakes in this process. However, we possess the capacity to employ context and non-visual cues to aid us in recovering from these errors.

Example 1. *Upon observing an individual, from the back and walking away from us, based on his gait and possibly body type, we tentatively conclude that the individual is Tom, a colleague at work. However, if we suddenly remember that Tom called in sick earlier in the day, we may decide that it cannot be Tom. Later still, if we observe that individual enter a Black BMW, a type of car we know Tom owns, we might conclude more strongly this time that it has to be Tom. However, before entering the car, if the individual turns around to face us and we realize that it is a person we have never seen before, we may definitely conclude that it is not Tom.*

The example demonstrates how humans employ common sense to reason about identities. Human reasoning is characterized, among other things, by [63]

1. **Its ability to err and recover** - This is important because when dealing with uncertain input, decisions or analysis made might have to be retracted upon acquisition of new information. In Example 1, we retracted our belief of the person being Tom or not several times,

2. **Its qualitative description of uncertainty** - a qualitative gradation of belief permits us to encode our confidence in decisions we make. In Example 1, our degree of belief in whether or not the person was Tom moved from slightly sure to definitely sure.

3. **Prioritization** - it is important to have a sense of how reliable our thread of reasoning is. In Example 1, based on appearance we were only slightly sure,

based on vehicle information we were more sure, based on face recognition we were definitely sure etc.

4.3 Reasoning Framework

Logic programming systems employ formulae that are either facts or rules to arrive at inferences. In visual surveillance, rules can be used to define various activities of interest as well as intermediate inferences such as that of equality of individuals. Rules are of the form "$A \leftarrow A_0, A_1, \cdots, A_m$" where each A_i is called an atom and ',' represents logical conjunction. Each atom is of the form $p(t_1, t_2, \cdots, t_n)$, where t_i is a term, and p is a predicate symbol of arity n. Terms could either be variables (denoted by upper case alphabets) or constant symbols (denoted by lower case alphabets). The left hand side of the rule is referred to as the head and the right hand side is the body. Rules are interpreted as "if body then head". Facts are logical rules of the form "$A \leftarrow$" (henceforth denoted by just "A") and correspond to the input to the inference process. These facts are the output of the computer vision algorithms, and include "atomic" events detected in video (entering/exiting a door, picking up a bag) and data from background subtraction and tracking. Finally, '$\neg$' represents negation such that $A = \neg\neg A$.

4.3.1 Default Logic

Logic programming based visual surveillance systems apply a set of predefined logical rules defining each activity to logical facts generated in real time from events transpiring in video to recognize activities [80]. Traditional logic programs are based on deduction, which is a method of exact inference. If the body of a rule evaluates to true, then the head always evaluates to true; in classical logic, there exists no provision of changing the truth value of the head over time. Deduction therefore requires information to be complete, precise and consistent. By contrast, in real world surveillance scenarios, one has to deal with incomplete, imprecise and potentially inconsistent information. Humans possess the ability to reason effectively under such circumstances using what is termed "common sense reasoning". Default logic [74] is an attempt to formalize common sense reasoning using default rules. Default logic expresses rules that are "true by default" or "generally true" but could be proven false upon acquisition of new information in the future. This property of default logic, where the truth value of a proposition can change if new information is added to the system, is called nonmonotonicity.

Definition 6 (Default Theory)**.** *A default theory Δ is of the form $\langle W, D \rangle$, where W is a set of traditional first order logical formulae (rules and facts) also known as the definite rules and D is a set of default rules of the form $\frac{\alpha:\beta}{\gamma}$, where α is known as the precondition, β is known as the justification and γ is known as the inference or conclusion.*

A default rule of this form expresses that if the precondition α is known to be true, and the justification β is consistent with what is currently in the knowledge base, then it is possible to conclude γ. Such a rule can be also written as $\gamma \leftarrow \alpha, not(\neg\beta)$. 'not' represents the negation by "failure to prove" operator and the consistency check for β is done by failure to prove its negation.

Example 2. *Assume the following set of rules and facts:*

$$\begin{aligned}
\neg equal(P_1, P_2) &\leftarrow distinct(P_1, P_2). \in W \\
equal(P_1, P_2) &\leftarrow appear\_similar(P_1, P_2), not(\neg equal(P_1, P_2)) \in D \\
&\quad\{appear\_similar(a, b)\}_t \\
&\quad\{appear\_similar(a, b), distinct(a, b)\}_{t+1}
\end{aligned}$$

where $\{\cdots\}_t$ *indicates the set of facts at time t and* $distinct(a, b)$ *indicates that a and b appear as two separate and distinct individuals at some point of time.*

In this example, at time t, given the rules and the set of facts, the system concludes that since it cannot prove $\neg equal(a, b)$ and $appear\_similar(a, b)$ is true, therefore $equal(a, b)$ is true. However, at time t+1, it is now possible to prove $\neg equal(a, b)$ because $distinct(a, b)$ is true and therefore the system now can no longer conclude $equal(a, b)$ (the default rule is blocked by the definite rule) and concludes $\neg equal(a, b)$ instead.

While the property of a conclusion blocking another default rule is desirable since it bestows nonmonotonicity upon the system, it can also create a problem.

Example 3. *Assume the following set of rules and facts:*

$$\begin{aligned}
\neg equal(P_1, P_2) &\leftarrow distinct(P_1, P_2), not(equal(P_1, P_2)). \in D \\
equal(P_1, P_2) &\leftarrow appear\_similar(P_1, P_2), not(\neg equal(P_1, P_2)) \in D \\
&\quad\{appear\_similar(a, b), distinct(a, b)\}_t
\end{aligned}$$

In Example 3, the rule for inferring that two individuals are not equal if they appear distinct is now made a default rule[1]. In this case, given the set of facts, at time t, depending on the order in which the default rules are applied, different sets of conclusions can be produced. If the first default is applied first, it blocks the second default and we conclude $\neg equal(a, b)$; but if the second default is applied first, it blocks the first and we conclude $equal(a, b)$.

Definition 7 (Extensions)**.** *The different sets of conclusions that can be derived by applying defaults in different orders are called extensions.*

A default theory can have multiple extensions, each capturing a possible outcome of the definite and default rules. While multiple extensions of a default theory list its possible outcomes, they are of not much use if a single solution is needed. There are several different approaches in the literature to obtain a single solution from the space of extensions of the default theory, including specificity [44], prioritized defaults [9] and multi-valued belief states [34]. Our system adopts the latter.

In the multivalued belief states approach, various rules in the system are regarded as different sources of information concerning the truth value[2] of a given proposition. These sources contribute different amounts of information to the decision making process and consequently our degree of belief in these propositions

[1] This default rule captures the fact that if there exists a mirror in the world, it could be possible for a single person to appear as two distinct individuals

[2] It is important to note that by truth value we mean our degree of belief in the veracity or falsity of a given proposition. This is different from the actual truth value of the proposition in the real world.

should mirror the information content. For example, default rules are not always correct and could be proven wrong by definite rules. Therefore, in this approach, definite rules provide more information than default rules. We seek a representation that combines truth value of these belief states with the information content of the sources.

4.3.2 Bilattice Theory

Bilattices [34] provide an elegant and convenient formal framework in which the information content from different sources can be viewed in a truth functional manner. Truth values assigned to a given proposition are taken from a set structured as a bilattice.

Definition 8 (Lattice)**.** *A lattice is a set L equipped with a partial ordering $\leq$ over the elements of L, a greatest lower bound (glb) and a lowest upper bound (lub) and is denoted by the triple (L,glb,lub) where glb and lub are operations from $L \times L \rightarrow L$ that are idempotent, commutative and associative*

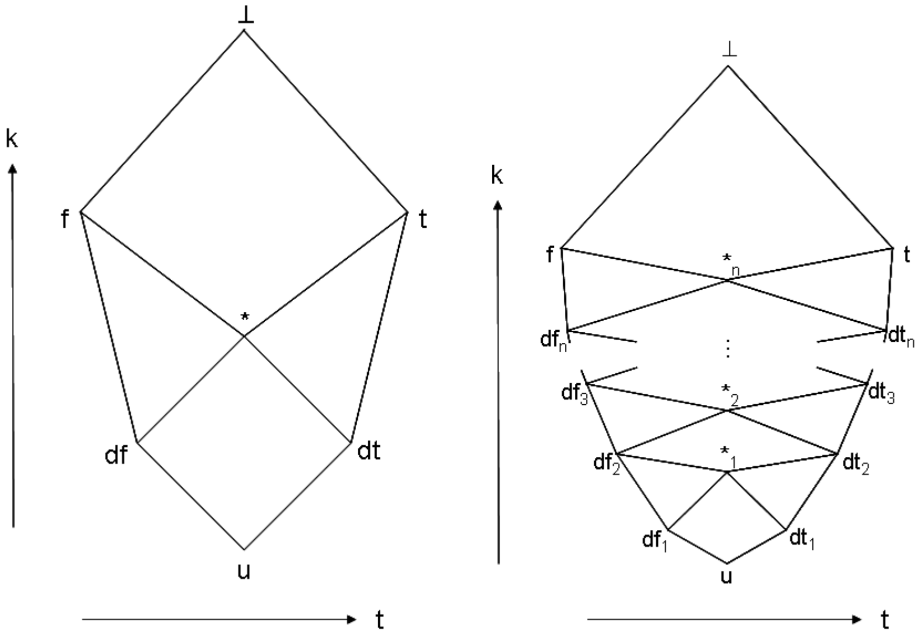
Informally a bilattice is a set, B, of truth values composed of two lattices $(B, \wedge, \vee)$ and $(B, \cdot, +)$ each of which is associated with a partial order $\leq_t$ and $\leq_k$ respectively. The $\leq_t$ partial order indicates how true or false a particular value is, with f being the minimal and t being the maximal. The $\leq_k$ partial order indicates how much is known about a particular sentence. The minimal element here is u (completely unknown) while the maximal element is $\perp$ (representing a contradictory state of knowledge where a sentence is both true and false). The glb and the lub operators on the $\leq_t$ partial order are $\wedge$ and $\vee$ and correspond to the usual logical notions of conjunction and distinction, respectively. The glb and the lub operators on the $\leq_k$ partial order are $\cdot$ and $+$, respectively, where $+$ corresponds to the combination of evidence from different sources or lines of reasoning while $\cdot$ corresponds to the consensus operator. A bilattice is also equipped with a negation operator $\neg$ that inverts the sense of the $\leq_t$ partial order while leaving the $\leq_k$ partial order intact.

Definition 9 (Bilattice [34])**.** *A bilattice is a sextuple* $(B, \wedge, \vee, \cdot, +, \neg)$ *such that*

- $(B, \wedge, \vee)$ *and* $(B, \cdot, +)$ *are both lattices and*
- $\neg$ *is a mapping such that*
  - $\neg^2 = 1$ *and*
  - $\neg$ *is a homomorphism from* $(B, \wedge, \vee)$ *to* $(B, \vee, \wedge)$ *and from* $(B, \cdot, +)$ *to itself.*

4.3.2.1 Properties of Bilattices

Figure 4.2(a) shows a bilattice corresponding to classical default logic. The set B of truth values contains, in addition to the usual definite truth values of t and f, dt and df corresponding to true-by-default (also called "decided-true") and false-by-default (also called "decided-false"), u corresponding to "unknown", * corresponding to "undecided" (indicating contradiction between dt and df) and $\perp$ corresponding to "contradiction" (between t and f). The t-axis reflects the partial ordering on



(a) (b)

Figure 4.2: (a)Bilattice for default logic (b) Bilattice for prioritized default logic.

the truth values while the k-axis reflects that over the information content. This bilattice provides us with a correlation between the amount of information and our degree of belief in a source's output. Procuring more information about a proposition, indicated by rising up along the k-axis, causes us to move away from the center of the t-axis towards more definitive truth values. The only exception to this being in case of a contradiction, we move back to the center of the t-axis. Negation corresponds to reflection of the bilattice about the $\perp -u$ axis. It is also important to note the this bilattice is distributive with respect to each of the four operators. Based on this framework, we can define the truth tables for each of the four operators as defined in figure 4.3.

$\wedge$	$\perp$	T	F	$*$	DT	DF	U
$\perp$	$\perp$	$\perp$	F	U	$\perp$	DF	U
T	$\perp$	T	F	$*$	DT	DF	U
F	F	F	F	F	F	F	F
$*$	U	$*$	F	$*$	$*$	DF	U
DT	$\perp$	DT	F	$*$	DT	DF	U
DF	DF	DF	F	DF	DF	DF	DF
U	U	U	F	U	U	DF	U

$\vee$	$\perp$	T	F	$*$	DT	DF	U
$\perp$	$\perp$	T	$\perp$	T	DT	$\perp$	T
T	T	T	T	T	T	T	T
F	$\perp$	T	F	$*$	DT	DF	U
$*$	T	T	$*$	$*$	DT	$*$	DT
DT	DT	T	DT	DT	DT	DT	DT
DF	$\perp$	T	DF	$*$	DT	DF	U
U	T	T	U	DT	DT	U	U

$\cdot$	$\perp$	T	F	$*$	DT	DF	U
$\perp$	$\perp$	T	F	$*$	DT	DF	U
T	T	T	F	$*$	DT	DF	U
F	F	F	F	$*$	DT	DF	U
$*$	$*$	$*$	$*$	$*$	DT	DF	U
DT	DT	DT	DT	DT	DT	DF	U
DF	DF	DF	DF	DF	DF	DF	U
U	U	U	U	U	U	U	U

$+$	$\perp$	T	F	$*$	DT	DF	U
$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
T	$\perp$	T	$\perp$	T	T	T	T
F	$\perp$	$\perp$	F	F	F	F	F
$*$	$\perp$	T	F	$*$	$*$	$*$	$*$
DT	$\perp$	T	F	$*$	DT	$*$	DT
DF	$\perp$	T	F	$*$	$*$	DF	DF
U	$\perp$	T	F	$*$	DT	DF	U

Figure 4.3: Truth table for glb and lub operators for t and the k axis of the bilattice for default logic.

4.3.3 Inference

Definition 10 (Truth Assignment)**.** *Given a declarative language L, a truth assignment is a function $\phi : L \rightarrow B$ where B is a bilattice on truth values.*

The semantics of a bilattice system is given by a definition of closure. If $\mathcal{K}$ is the knowledge base and ϕ is a truth assignment labelling each sentence $k \in \mathcal{K}$ with a truth value then the closure of ϕ, denoted $cl(\phi)$, is the truth assignment that labels information entailed by $\mathcal{K}$. For example, if ϕ labels sentences $\{p, q \leftarrow p\} \in \mathcal{K}$ as true; i.e. $\phi(p) = T$ and $\phi(q \leftarrow p) = T$, then $cl(\phi)$ should also label q as true as it is information entailed by $\mathcal{K}$. Entailment is denoted by the symbol '$\models$' ($\mathcal{K} \models q$).

If $S \subset L$ is a set of sentences entailing q, then the truth value to be assigned to the conjunction of elements of S is

$$\bigwedge_{p \in S} cl(\phi)(p) \tag{4.1}$$

This term represents the conjunction of the closure of the elements of S. It is important to note that this term is merely a contribution to the truth value of q and not the actual truth value itself. The reason it is merely a contribution is because there could be other sets of sentences S that entail q representing different lines of reasoning (or, in our case, different rules). The contributions of these sets of sentences need to be combined using the + operator. Also, if the expression in 6.2 evaluates to false, then its contribution to the value of q should be "unknown" and not "false". These arguments suggest that the closure over ϕ of q is

$$cl(\phi)(q) = \sum_{S \models q} u \vee [\bigwedge_{p \in S} cl(\phi)(p)] \tag{4.2}$$

We also need to take into account the set of sentences entailing $\neg q$. Since $\phi(\neg q) = \neg\phi(q)$, aggregating this information yields the following expression

$$cl(\phi)(q) = \sum_{S \models q} u \vee [\bigwedge_{p \in S} cl(\phi)(p)] + \neg \sum_{S \models \neg q} u \vee [\bigwedge_{p \in S} cl(\phi)(p)] \tag{4.3}$$

For more information on the properties and logical inference based on bilattice theory see [34].

Example 4 (Inference example)**.**

$$\begin{aligned}
\phi[\neg equal(P_1, P_2) \leftarrow distinct(P_1, P_2)] &= DT \\
\phi[equal(P_1, P_2) \leftarrow appear\_similar(P_1, P_2)] &= DT \\
\phi[appear\_similar(a, b)] &= T \\
\phi[distinct(a, b)] &= T
\end{aligned}$$

$$\begin{aligned}
cl(\phi)(equal(a, b)) &= [U \vee (T \wedge DT)] + \neg[U \vee (T \wedge DT)] \\
&= [U \vee DT] + \neg[U \vee DT] = DT + DF = *
\end{aligned}$$

In Example 4, we encode our belief that the two rules are only true in general and do not always hold by assigning a truth value of DT to them. We record our belief in the facts as T and apply equation 6.4 to compute the truth value of $equal(a, b)$. Note in Example 3, we obtained two extensions with $equal(a, b)$ being true in one and $\neg equal(a,b)$ being true in another. Using the multivalued logic approach we collapse these extensions and combine the two conclusions to obtain $DT + DF = *$ or “undecided”.

4.3.4 Belief Revision

In classical AI, belief revision is the process of revising a proposition’s belief state upon acquisition of new data. In the bilattice framework presented above, these revisions should only occur if the new data source promises more information than that which triggered the current truth value assignment. Note that the belief combination operator, + is a lub operator on the k-axis, meaning it will only choose a sentence with maximum information.

However, this poses a problem for our current theory. Since default rules could be contradicted by other default rules, it is possible that many propositions will suffer from a DT, DF contradiction and will settle in the * or undecided state. According to our current theory, only a rule with more information, the definite rules, can release it from this state. Unfortunately in visual surveillance, most rules are default rules and therefore it might be the case that there may be no definite rules to rescue a proposition once it gets labelled “undecided”.

Example 5. *Assume that an individual enters a room we believe to be empty and closed (no other exit). Assume also that after some time, another individual emerges from the room who appears dissimilar from the first individual*

$$\begin{aligned}
\phi[\neg equal(P_1,P_2) \leftarrow \neg appear\_similar(P_1,P_2)] &= DT \\
\phi[equal(P_1,P_2) \leftarrow enterclosedworld(P_1,X,T_1), & \\
exitclosedworld(P_2,X,T_2), T_2 > T_1, & \\
emptybefore(X,T_1), emptyafter(X,T_2), & \\
not(enter\_or\_exit\_between(P_3,T_1,T_2)).] &= DT
\end{aligned}$$

$$\begin{aligned}
\phi[\neg appear\_similar(a,b)] &= T \\
\phi[enterclosedworld(a,office,400)] &= T \\
\phi[exitclosedworld(b,office,523)] &= T \\
\phi[emptybefore(office,400)] &= DT \\
\phi[emptyafter(office,523)] &= DT \\
\phi[not(enter\_or\_exit\_between(P_3,400,523)] &= T
\end{aligned}$$

$$cl(\phi)(equal(a,b)) = \cdots = DT + DF = *$$

In Example 5, the first rule states that if two individuals do not appear similar, then they are not equal. The second rule, states that if there exists a closed world that we believe to be empty and we observe an individual enter it and at a subsequent time exit it and no one else has entered or exited the closed world in between, then we can conclude that the two individuals are equal. The set of facts captures the activity of an individual entering a closed empty world and later reappearing and looking dissimilar from the individual who entered. In this case, too, we have contradicting defaults and on applying equaltion 6.4, $equal(a, b)$ gets labelled *.

4.3.5 Prioritized Defaults

This problem arises because thus far we are assuming that all the default rules provide us the same amount of information, causing them to contradict each other and force a proposition into the * state. However, suppose, instead we assume that different defaults could provide different amounts of information and consequently could alter our belief state by different degrees. It turns out that the bilattice structure very elegantly generalizes to accommodate this assumption. We could modify the previous example and state that inferring equality based on appearance matching is a weaker default than inferring equality based on the fact that the person entered and exited an empty closed world. Therefore, if we then assign a label DT_1 to default 1 and label DT_2 to default 2 and state that DT_2 is a stronger default and has more information than DT_1 we can conclude

$$cl(\phi)(equal(a, b)) = DT_2 + \neg DT_1 = DT_2 + DF_1 = DT_2$$

Figure 4.2(b) shows a general bilattice for a prioritized default theory with n priorities. Formally a prioritized default theory $\Delta_<$ is of the form $\langle W, D, < \rangle$ [9] where W and D are as defined in Definition 6 and $<$ is a strict partial ordering on D. The semantics of the bilattice on the new set of truth values stays the same as before.

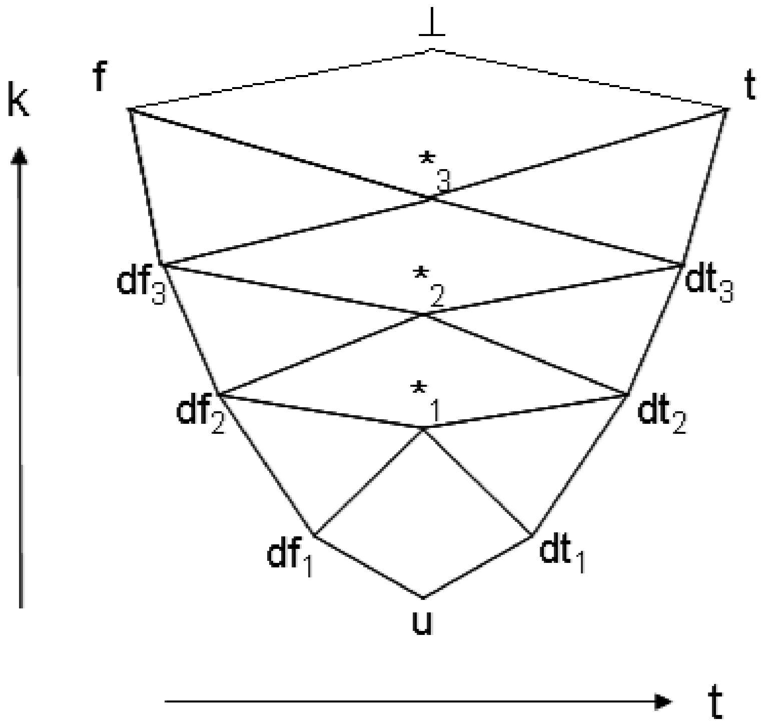


Figure 4.4: Prioritized bilattice employed in our system

4.4 Reasoning about Identities

Our system primarily employs four identifying cues or traits for reasoning about identities. These cues are based on the individuals possessions, closed world activity, knowledge and appearance. In addition to these cues, we also employ equality axioms of reflexivity, transitivity, and symmetry.

Identity can be verified on basis of a person possessing something that only he can possess. For example, if we know that a vehicle belongs to an individual and later we observe another individual entering that vehicle using a key, we can conclude that they must be equal. An individual can be identified on the basis of certain closed world activities, examples of which we have seen earlier (see Example 5). One can also verify identity on the basis of the knowledge we think an individual possesses. For example, if there is a combination lock on a door controlling access to a office and we observe an individual successfully entering the code and opening the door to enter the room, we can conclude that he must be the owner of that office. Finally appearance based cues help identify individuals based on appearance. We employ a color histogram based appearance matching algorithm.

It should be noted that any rule based on these cues can almost never be definitive and most of them will be default rules. Also, different cues provide us with different amounts of information as they deal with varying degrees of uncertainty. Without loss of generality, we assume three levels or priorities of defaults[3]. Also, we assume that the definite rules are never incorrect and therefore there will never occur a contradiction between T and F. Figure 4.4 shows the resultant bilattice employed in our system.

4.4.1 Rules of identity

In this section we will give English descriptions of various rules employed in our system, and note their priority levels.

[3] The number of levels depend on the number and type of default rules. In our system and for the environment we are observing as we shall see in subsequent sections, there is no justification to have more than three levels.

4.4.1.1 Priority Level 1

Appearance based identification states that if two individuals appear similar to each other then they are equal to each other. On the other hand, if two individuals do not appear similar to each other, then they are not equal. These set of rules are required in situations where we are forced to compare individuals in the absence of any contextual information. Assume an individual disappears from view into an open world (a world with no constraints on the movements of that individual or others) and another person reappears. Since the person reappearing could potentially be anyone in the world, there is significant uncertainty associated with making an identity decision. Therefore, these rules provide us with least information compared to any approach that augments appearance matching with context. We therefore assign to it priority level 1[4]

4.4.1.2 Priority Level 2

If a number of individuals are observed entering a closed world and later reappearing, the uncertainty associated with performing appearance matching as before on that limited group of people is significantly lesser than in the previous case. Therefore, this rule, which reduces the space of possible matches via a closed world assumption, provides more information than pure appearance matching and we as-

[4] Note, in our system we employ color histogram based appearance matching which by itself is a poor biometric, however if one were to employ a more powerful biometric system such as fingerprint recognition or even high resolution face recognition, then such a cue would possibly figure higher up in the bilattice.

sign to it priority level 2.

4.4.1.3 Priority Level 3

Most of the rules based on possession and knowledge fall in this category as they cause us to depart from comparing groups of individuals to comparing just two individuals. For example, if we observe an individual arrive in the scene in a vehicle, disappear from view and subsequently another individual appears in the scene and uses a key[5] to enter the vehicle, we can conclude, provided they appear similar, that they must be equal. Here we are comparing just two individuals the one who arrived in the vehicle and the one departing in it. Similar reasoning can be applied to offices which require a key or a combination number to enter[6]. Since the comparisons here involve an even more reduced set than the previous case, we assign to this set of rules priority level 3

Another set of rules that fall in this prioritization are purely closed world based rules such as an individual entering a closed world that we believe to be empty and subsequently exiting it such that no other individual is observed entering or exiting the closed world in between. Here, since there exists the possibility of the individual changing their attire inside the closed world (taking off a jacket),

[5] At present we do not directly recognize an action like using a key. Also, many vehicles have remote door locks which do not require a physical key. The fact that the individual uses a key is a default assumption. We assume that if the individual purposefully walks to the vehicle and enters it, he probably has a key. This is in contrast to loitering around the car for a while or moving from car to car, and then entering one.

[6] provided we have reason to believe that the office usually has only one occupant

appearance matching is not a strong cue. Other rules in this category are rules that state that if we observe an individual enter a closed world and if, while we believe he is still inside, we observe another individual elsewhere in the scene, then they cannot be equal to each other. Closed world rules such as these clearly have more information than rules with priority levels 1 and 2; however it isn't clear that they have more or less information than the knowledge and possession based rules mentioned above. Therefore we assign to these set of rules priority level 3.

4.4.1.4 Definite rules

It is very hard to state that two individuals are definitely equal based on visual observation alone. Irrespective of how much information one packs in such rules, it is always possible to find ways to defeat them. Therefore, in our system we do not have a single rule that definitely infers equality. However, it is possible to state that two individuals are not equal. We do that when we observe them as two distinct individuals at the same instant of time[7]. We also consider the equality axioms of reflexivity, transitivity and symmetry to be definite in nature.

4.5 Activity Recognition

We can now use inferences made regarding equality of individuals to reason about the occurrence of various activities in the input video. Moreover we can propagate our degree of belief in the identity statement to the activities that it

[7] The assumption is there are no mirrors in our world. Reflective surfaces such as glass windows never act like true mirrors, thereby giving the individual's reflection a different appearance

contributes to. We define three such activities and list some sample rules[8].

4.5.0.5 Theft:

We define theft as the activity of an individual possessing a package that does not belong to him. A package does not belong to an individual P_1 at time T_1 if it belonged to another individual P_2 at some time $T_2 < T_1$ such that $\neg equal(P_1, P_2)$. Formally,

$$\begin{gathered} theft(P_1, B, T_1) \leftarrow human(P_1), bag(B), possess(P_1, B, T_1), \neg belongs(B, P_1, T_1). \\ \neg theft(P_1, B, T_1) \leftarrow human(P_1), bag(B), possess(P_1, B, T_1), belongs(B, P_1, T_1). \end{gathered} \tag{4.4}$$

A package does not belong to an individual P_1 at time T_1 if it was originally possessed by individual P_2 at some time $T_2 < T_1$ such that $\neg equal(P_1, P_2)$.

$$\begin{aligned} \neg belongs(B, P_1, T_1) &\leftarrow original\_possessor(P_2, B, T_2), T_2 < T_1, \neg equal(P_1, P_2). \\ belongs(B, P_1, T_1) &\leftarrow original\_possessor(P_2, B, T_2), T_2 < T_1, equal(P_1, P_2). \end{aligned}$$

[8] Note, due to space constraints, rules listed in this chapter are only those pertinent to the scenarios described in the next section and represent a small (modified for ease of understanding) subset of the rules encoded in the system. Typically for any predicate p, there exist multiple rules deriving p and/or $\neg p$ depending on how we want the system to behave under various scenarios.

4.5.0.6 Entry Violation:

Assuming an identity card reader controls access to a building entrance, we define entry violation as the activity of an individual entering the building without scanning his card. Formally,

$$\neg entry\_violation(P_1) \leftarrow enter(P_1, T_1), scancard(P_2, T_2), T_2 < T_1, equal(P_1, P_2).$$

$$entry\_violation(P_1) \leftarrow enter(P_1, T_1), scancard(P_2, T_2), T_2 < T_1, \neg equal(P_1, P_2).$$

4.5.0.7 Unattended Package:

We define a package to be unattended if we observe an individual drop off a package and then cease to be in its vicinity. This is captured by the following rules

$$\neg unattended(B, T_1) \leftarrow in\_vicinity(P_1, B, T_1), dropoff(P_2, B, T_2), equal(P_1, P_2).$$

$$unattended(B, T_1) \leftarrow not(\neg unattended(B, T_1)).$$

Propagation of belief states from equality statements to these activities is done using equation 6.4.

4.6 Experiments

The proposed framework has been built on top of VidMAP presented in chapter 3. Multivalued default reasoning is implemented using meta-predicates provided by Prolog. As currently implemented, this application runs at frame rate while taking input from up to three cameras.

The application consists of two kinds of threads: the (possibly multiple) camera thread(s) which take input from the camera(s) and detect “atomic” events (like entering a door or picking up a bag) and a single reasoning thread responsible for the high level multivalued default reasoning. For each camera connected to the system, we create a camera thread that first performs background subtraction and tracking on the video. It then detects “atomic” events and syntactically structures them as Prolog facts. The reasoning thread, when first created, starts the Prolog engine and initializes it by inserting into its knowledge base all the predefined rules from the default theory. The reasoning thread is subsequently evoked every few seconds. Every time it runs, it assimilates Prolog facts generated by the camera threads and inserts them into the Prolog engine’s knowledge base. Also, for every human observed in the video, it reasons about their identity by applying all applicable equality rules. Finally, equality statements, along with their qualitative confidence values, are used to reason about the occurrence of predefined activities using the rules listed in section 4.5. If any of the activities can be proven with belief states of DT_1, DT_2, DT_3 or T then the reasoning thread generates an alert.

The tool we have built also allows the user to manually click on the image, while setting up the system, to mark and label regions (as ‘closed world’, ‘hand-off region’, ‘card reader’ etc.), in the scene. These regions, as seen in Fig 4.5 and 4.6 provide the system with information about the scene structure and properties and also helps the system to recognize a richer set of “atomic” events that log the interactions of individuals with the environment.

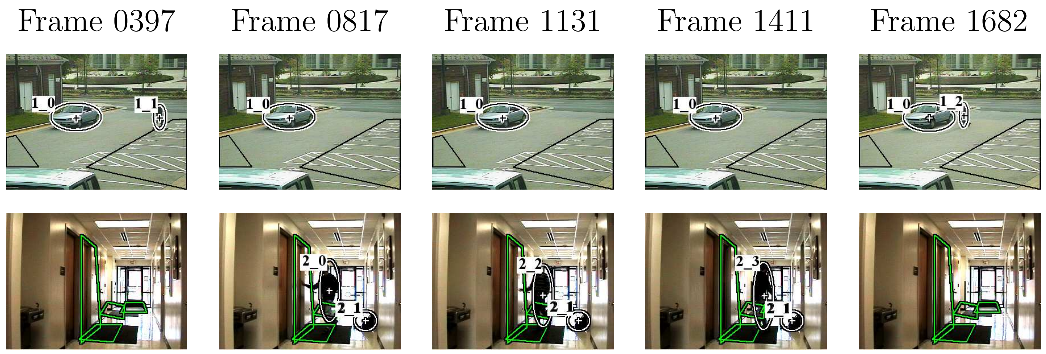


Figure 4.5: Figure depicting scenario 7. Top row Camera 1 and bottom row Camera 2

4.6.1 Scenarios

We demonstrate our system in action on a multi-camera surveillance setup. We employ cameras that have disjoint fields of view and label certain regions within the scene as hand-off regions. Hand-off regions are areas within an image where individuals disappear and reappear between cameras. We encode simple rules that state that if an individual disappears from the hand-off region in one camera and within a certain time interval appears within a specific hand-off region of another camera and the two individuals appear similar, then they must be equal. These rules as well as the belief states assigned to them are clearly setup specific.

We now describe a few scenarios that were used to test the system and describe how the system performed.

Scenario 4 (Theft-See Figure 4.5 and supplemental video)**.** *Vehicle* 1_0 *enters the scene and individual* 1_1 *appears from it and disappears from the view of camera 1 from the right hand-off region. He appears in view of camera 2 from its hand-off region as* 2_0*, drops a bag,* 2_1*, in the corridor and enters a room (closed world).*

He is followed by another individual 2_2 *(who appears from around the corner) into the room. Subsequently an individual* 2_3 *exits the room, picks up the bag and exits the view of camera 2 through the hand-off region. He appears in the hand-off region of camera 1 as* 1_2 *and enters the vehicle using a key and drives away.*

In this scenario, the system correctly identifies human 2_0 as being equal to 1_1 due to the hand-off rules encoded for this camera setup. When human 2_3 exits the room, the system attempts to apply the closed world and appearance matching (default priority 2) set of rules mentioned in section 4.4. However, it turns out 2_3 appears similar to both 2_0 and 2_2, and therefore the system derives both $\phi[equal(2\_3, 2\_0)] = DT_2$ and $\phi[equal(2\_3, 2\_2)] = DT_2$. Note the system can also prove $\phi[equal(2\_0, 2\_2)] = DF_3$ which is inconsistent if we attempt to establish the transitivity relation. The system therefore is forced to assign $\phi[equal(2\_3, 2\_0)] = *_2$ and $\phi[equal(2\_3, 2\_2)] = *_2$ which represents the undecided state. When 2_3 picks up the bag left behind by 2_0, the system tries to prove whether or not a theft has taken place, however, it can only prove $\phi[theft(2\_3, 2\_1, 1415)] = *_2$ due to the uncertainty involved in the equality statement that contributes to it. The system continues on to correctly conclude that human 2_3 is equal to human 1_2. However, when 1_2 uses a key and enters the vehicle, it can now prove $\phi[equal(1\_1, 1\_2)] = DT_3$. By transitivity, the system is then able to revise its belief of $\phi[equal(2\_3, 2\_0)]$ from $*_2$ to DT_3 and consequently revise its belief of $\phi[theft(2\_3, 2\_1, 1415)]$ from $*_2$ to DF_3, i.e. no theft has occurred with high confidence.

In the next scenario, we assume there exists a card reader controlling access

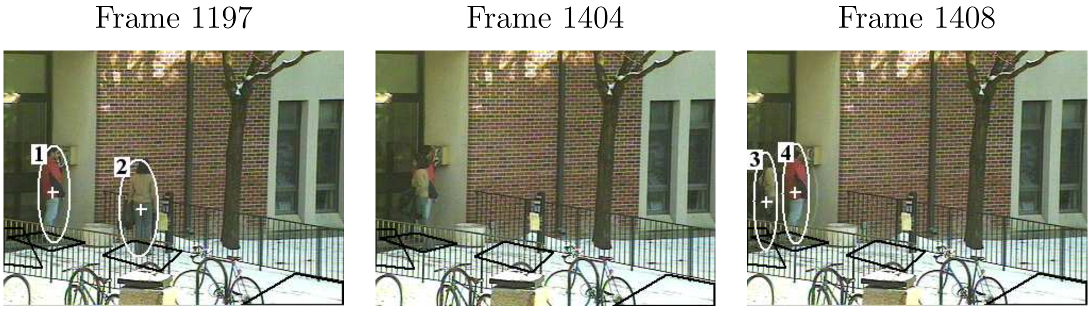


Figure 4.6: Figure depicting scenario 8.

to a building.

Scenario 5 (Entry Violation)**.** *Individual 2 approaches the card reader and swipes her card while 1 is at the phone. Individuals 1 and 2 momentarily occlude each other causing the tracker to lose track of the individuals. Subsequently when the two individuals separate out again, tracking is resumed and human 3 enters the building.*

In this scenario, after tracking is lost and resumed, the system needs to ascertain whether the person who entered the building is the one who swiped the card. However due to a lack of any context based cues, it is forced to resort to appearance matching (priority level 1) rules. Based on those rules, the system concludes $\phi[equal(2,3)] = DT_1$ and $\phi[entry\_violation(3)] = DF_1$, i.e. no entry violation has taken place with low confidence.

Scenario 6 (Unattended Package)**.** *Human* 2_16 *drops a bag* 2_17 *in the corridor and enters an empty room (closed world). Subsequently* 2_18 *exits the room.*

In this scenario, the event of 2_16 entering the room, triggers the unattended package alert as the bag's owner is no longer in its vicinity. However, when 2_18 appears, based on the closed world (priority level 3) rules, the system is able to conclude $\phi[equal(2\_16, 2\_18)] = DT_3$ and therefore it also concludes $\phi[unattended(2\_17, 1783)] = DF_3$, i.e. the bag is not unattended with high confidence.

4.6.2 Complexity

Traditional default logics are computationally intractable. In traditional default logic, inferences can only be made if they are consistent with the current knowledge base. Consistency checks in default logics are a primary source of intractability and are required because the traditional theory does not permit inconsistent information to persist. In our framework, however, since the truth values are really only an agent's belief state about the world, we relax the consistency condition and allow for seemingly contradictory information to persist. Our framework therefore avoids explicit consistency checks.

Another source of intractability for traditional default logics is the method of choosing a consistent set of propositions entailed by the default theory from the set of all its extensions. Regardless of what technique is adopted to achieve this, enforcing the consistency constraint requires one to generate and inspect all possible extensions of the default theory. Note, given n defaults, there are potentially $n!$ extensions that need to be examined. We avoid this source of intractability, again, by relaxing the consistency constraint and believing everything our theory tells us (albeit with different degrees of belief). The effect this achieves is that different extensions of our default theory are collapsed into a single solution. This makes sense because we treat our default rules as different sources of information, none of which can be completely discarded, and combine them in an information centric manner. A more disciplined and formal analysis of the complexity of the proposed theory is part of our future work.

4.7 Summary

The problem of identity maintenance is a very important problem in visual surveillance. Many activities that we wish to recognize in surveillance video depend, in some ways, upon the identities of the individuals involved, and therefore have to account for the uncertainty in reasoning about them. Traditionally, identity maintenance has relied solely on appearance matching, however it is extremely important to take into account context and cues provided by certain self-identifying actions to augment reasoning. This work is an attempt to provide a framework to do just that. The development of this framework has been heavily influenced by human reasoning. We believe human reasoning is characterized, among other things, by nonmonotonicity, qualitative belief gradation and prioritization. We have attempted to capture these traits in the proposed theory.

Chapter 5

Top-Down, Bottom-Up Reasoning for Occlusion Handling

5.1 Introduction

Automated visual surveillance systems require the capacity to detect and persistently track objects from their point of entry into the field of regard of the cameras, to their point of exit. This is, however, an extremely difficult task as object tracks are often lost due to occlusions by static structures in the scene or interactions with other tracked objects. Figure 5.1 shows individual 0 and individual 1 disappearing from view and subsequently individual 3 appearing in view from behind an occlusion. A persistent tracking system should be able to determine whether $individual(3) = individual(0)$ or $individual(3) = individual(1)$ and resume tracking. Moreover, if multiple individuals interact, then regardless of what goes on during the interaction event, when the individuals separate out, the system should be able to correctly establish identity of all individuals involved.

Traditionally, the problem of identity maintenance after occlusions has been handled by appearance matching. The basic premise in such approaches is that if two individuals appear similar to each other than they must be equal, while if they appear dissimilar, then they must be not equal. However, it is additionally possible to employ context based cues to perform identity maintenance. For example, in Figure 5.1, individual 3 drops a bag after emerging from the occlusion. Under certain

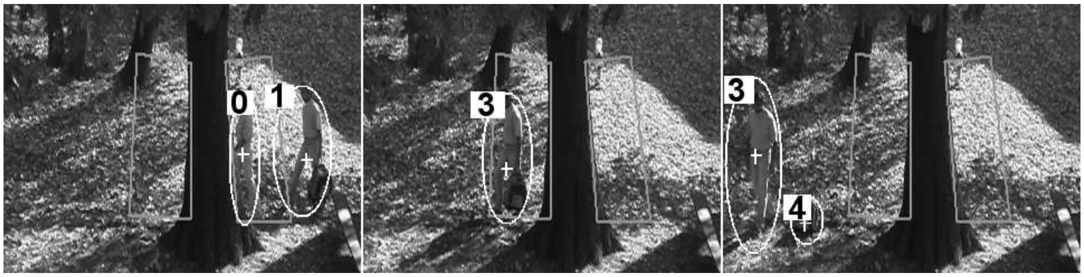


Figure 5.1: Sequence of images showing individual 0 and 1 disappearing from view and 3 subsequently appearing from behind an occlusion

circumstances, it might be possible to conclude that $individual(3) = individual(1)$ based on the fact both of them are carrying a similar looking bag. However, it is generally difficult to detect that either of 0 or 1 was carrying a bag. The fact that 3 was carrying a bag would be discovered first (e.g. through background subtraction) because a “bag-like” object, 4, splits from 3. Given this information, and assuming that the bag 4 did not change hands during the short visibility gap, searching for the image of bag 4 in the archived images of 0 and 1 can lead us to conclude that $individual(3) = individual(1)$.

In most vision systems the flow of information is usually bottom-up. The low level computer vision routines are run first to gather information which is then provided to high level reasoning routines. However, the situation described above requires information to flow top down. The reasoning module has to understand that there exists a deficit of information (perhaps because appearance matching by itself was unable to distinguish between the individuals in Figure 5.1) and given that 3 was carrying a bag, try to actively search archival video for the presence of a bag in images of individuals 0 or 1. This example captures the general problem of context driven image analysis that we address here.

In chapter 4 and [81], we described a system to perform identity maintenance across possibly large visibility gaps by employing, in addition to traditional appearance matching, several context based cues. We proposed a multivalued default logic (MVDL) framework in which these cues were regarded as different sources of information regarding the truth value of a given equality statement and were integrated in an information centric manner. This system recognized the occurrence of low level "atomic" events from the input video and provided this information to the high level MVDL reasoning framework where identity decisions were made based on low level observations. The flow of information was strictly bottom-up.

Here, we describe a system that treats occlusions and object interactions as closed world events and uses the MVDL framework to explicitly reason about object identities upon re-appearance. Specifically our contribution is the use of the high level reasoning framework to actively resolve states representing contradictions or lack of information regarding equality of individuals, by providing control feedback, and driving low level image processing modules.

5.2 Overview

Persistent tracking systems have to contend with two kinds of events. The first kind is when a tracked object is occluded by a static scene structure such as a tree or a pillar and the second kind is when two or more tracked objects interact, visually merging into one. When multiple tracked objects interact, our system does not attempt to separate out the individual objects that make up the merged region. It, instead, maintains in its knowledge base that, for the duration of the interaction event, the region being tracked is composed of multiple objects. Henceforth, occlusions caused both by static structures in the scene as well as object interactions will be collectively referred to as occlusion events.

As mentioned earlier, in addition to appearance matching, there exist several identifying cues that can provide information about an individual's identity. Knowledge about these cues and object behavior is encoded in our system as rules in a logic programming language. Since this knowledge represents behavior of objects in a real world, it can never be definite and completely correct. We therefore employ default logic (which is the core of the MVDL framework) as the language to specify these rules, which provides our framework the important property of nonmonotonicity (the property of retracting or disbelieving old beliefs upon acquisition of new information). Default rules capture information that are "true by default" or "generally true" but may cease to be true in the future.

The MVDL framework bestows upon the system, in addition to the property of nonmonotonicity, the capacity to qualify identity decisions with a qualitative confidence measure. Identity rules included in the MVDL framework are applied to reason about object identity when an object appears from occlusion events. These rules model occlusion events as closed world spaces; meaning that an object that enters an occlusion event should (with exceptions) exit from it eventually. Default logic models such assumptions while maintaining the possibility for it to be incorrect. If only a single object is involved in an occlusion event, such as an individual walking behind a tree and later reappearing, the uncertainty involved in making an identity decision is the least (as compared to scenarios where multiple objects are involved) and therefore, such a default rule will have a very high priority (or certainty) in the MVDL framework. This is in contrast to a situation where multiple individuals are involved in an occlusion event and the system is forced to establish identity solely on the basis of appearance matching. Identity decisions made on the basis of solely appearance matching typically have the lowest priority.

Unfortunately, however, it is often the case that the only information the system has access to is the appearance matching score. In such cases, the system might quickly enter a state of no information or contradiction regarding identity of a few individuals. To emerge from this state, the system tries to actively search for specific objects of interest that will make identification more certain. The system is composed of two layers, with the low level module performing object detection, local low level tracking and fact generation, and the high level module responsible for identity maintenance and providing control feedback to the low level module for conducting searches focused on archival video.

5.3 Low Level Module

We formulate various rules for object equality and inequality based on our knowledge about scene structure and behavior of humans, vehicles and packages. These rules need to be applied to logical facts. Logical facts are generated by the system by taking in video and recognizing the occurrence of certain ground atomic events, such as an individual entering or exiting the field of view or dropping or picking up a package.

5.3.1 Object Detection and Local Tracking

Surveillance setups typically consist of cameras that are either fixed and observe the same scene at all times or cameras that can perform pan-tilt-zoom operations. Assuming static surveillance cameras gives us the advantage of being able to employ binary information obtained from background subtraction to detect objects of interest. We employ a background subtraction algorithm proposed in [55]. Tracking of these objects is performed by detecting foreground "blobs" in each frame and then matching them across consecutive frames using their color and spatial information. Across consecutive frames, these blobs could either continue to persist, merge with or split from other blobs, appear or disappear. We are exclusively concerned with tracking only three kinds of objects - human, vehicles and packages (as opposed to tracking an arbitrary object like a hand). Due to a variety of reasons, background subtraction routinely introduces artifacts that can get tracked and erroneously labelled as objects of interest. Filtering out such noisy data is important for any tracking system; we do this by observing whether a blob continues to persist across several frames or not.

While this form of temporal filtering culls out isolated blobs that might appear due to background subtraction errors, it does not remove regions comprised of pixels that deviate from the background model due to physical interactions between humans, vehicles, or packages and the scene. Examples are reflections and shadows that appear disconnected from the shadow-casting object. Filtering out of these artifacts can be assisted by the application of knowledge about the behavior of humans, vehicles and packages.

5.3.2 Fact Generation

Facts are generated when certain atomic events occur in the video including when objects appear, disappear, merge and split. These facts are annotated with named regions in the scene where they occur. These regions are manually labelled at setup. This helps us to generate relevant facts when an object interacts with that region in the image. From the point of view of tracking, we are primarily interested in regions in the image where we expect objects to appear and disappear from view. These regions typically correspond to scene boundaries, visible portals (doors to open or closed worlds) and static structures in the scene which could potentially occlude tracked objects. Objects could also appear or disappear from the scene in areas other than the ones listed above. This usually happens when objects merge with or split from other tracked objects.

Objects can be occluded by static structures in the scene such as trees, pillars, boards etc. by moving behind them. We consider these occlusions to be closed worlds and expect that the objects will eventually emerge from them. An object disappearing into a closed world will cause the system to generate a fact $disappear(X, Closed\_World, T)$ where X is the object identifier, $Closed\_World$ is the identifier for the occlusion region in question and T is the time of the event. Similarly when an object appears from a closed world, a corresponding *appear* fact will be generated. Tracked object interactions are also regarded as closed worlds and similar appear and disappear facts are generated when an object merges with or splits from another object.

Facts are also generated to record appearance matching scores between individuals observed in video. We employ two kinds of appearance matching algorithms. The first is a simple color histogram based algorithm. A histogram is constructed from the color values of the pixels in a segmented object and compared against color histograms of other objects using the Bhattacharyya distance. The second appearance matching algorithm is more sophisticated and is run using the feedback control mechanism described in the next subsection.

5.3.3 Control Feedback

When the high level module detects a deficit of information, it directs the low level module to specifically gather information from certain space-time locales in the video. We maintain a queue of the previous 300 seconds of video along with the corresponding tracking data. This information also includes the segmentation information for each object detected.

Feedback in our system is of three types: (a) tracking back in time (b) specifically searching for an object attached or contained within another object and (c) matching appearances of two individuals using spatial and color information. If the high level module requires tracking backward in time, it sends a "track_back" request to the low level module with the identifier of the object to be tracked and the time from which the tracking is to be done. We employ a mean-shift [15] based tracker which is initialized on the object to be tracked at the point it is detected and run backward in time. A matching score is maintained and analyzed at each frame to ascertain whether or not the object being tracked is being tracked reliably. Tracking stops if either the queue of archived frames is exhausted or the matching score drops below a certain threshold. In either case, the last tracked location is reported to the high level module.

If continuous tracking back in time is not possible e.g. if the target object itself is occluded as shown in Figure 5.1, then the high level module sends a "search" request with the frame number for searching and the identifier for the object of interest. Searching for an object attached or contained within the image of another object is performed using the standard cross correlation based hierarchical image matching algorithm. The search is carried out for 5 frames before and after the search frame requested by the high level module to avoid erroneous matches as far as possible.

The high level module can also request a higher complexity appearance matching between two individuals. It does this by sending a "match" request to the low level module which contains identifiers for both individuals it wants a matching score for. This appearance matching algorithm is the color path length based approach [96], which combines color and a geodesic path length measure within a person's body to construct a statistical appearance model. Models are compared using the Kullback distance. The path length for a pixel is the shortest distance measured along a path lying within the body, from the head to that pixel. See Figure 5.2, which displays the paths along which this distance is computed for a hand pixel. Note that although the Euclidean distance between the hand pixel and the head is different for the two poses, the geodesic distance stays nearly the same.

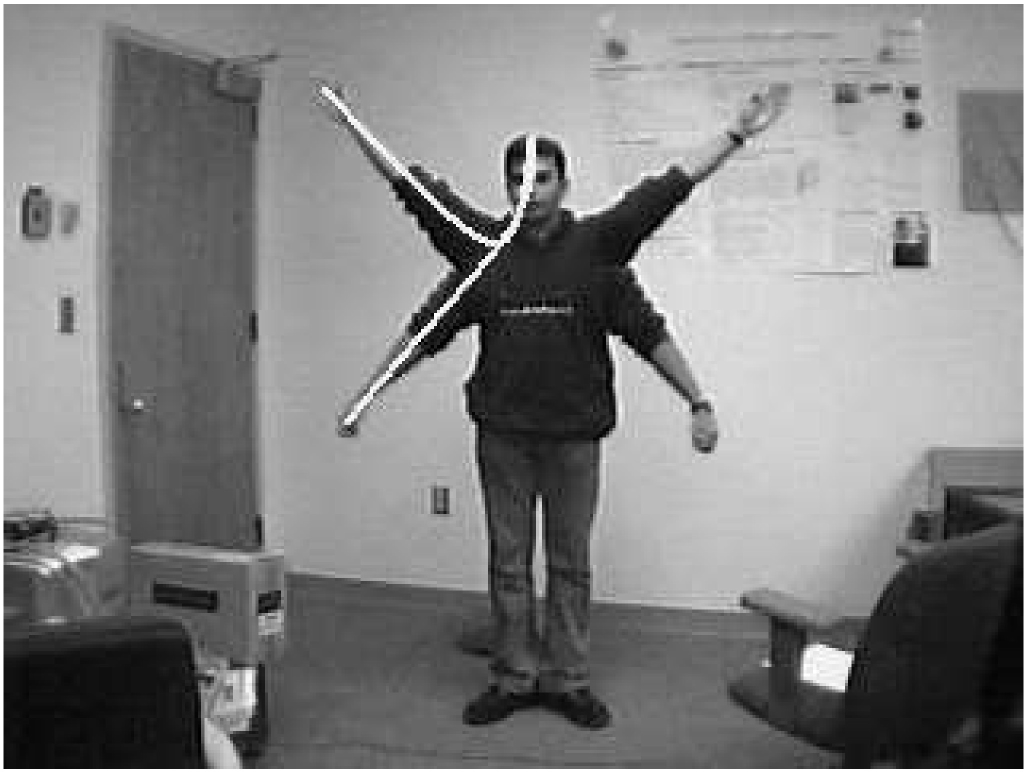


Figure 5.2: Figure showing geodesic path length for a hand pixel as measured from the head for two different poses.

5.4 High Level Module

The primary task of the high level module is to reason about an object's identity when it appears from occlusions. It does so both in the bottom-up fashion, by taking input from the low level module and processing it as well as in the top-down manner, actively seeking information where deemed necessary. Reasoning is performed by applying predefined identity rules formulated in the MVDL framework to the set of facts generated by the low level module. When the truth value assigned to an identity decision is either unknown (u) or undecided ($*_{1 \cdots n}$), the high level system determines to see if there exist any contextual cues that it can exploit. If they do then, it provides control feedback to the low level module and directs it to collect historical information that will help it emerge from the unknown or undecided states of belief.

5.4.1 Reasoning about identity

In [81], we employed four identifying cues or traits for reasoning about identities. These cues are based on the individuals possessions, closed world activity, knowledge and appearance. We continue to employ these identifying cues, although in a slightly different manner. In [81], we were primarily concerned with establishing identity across large visibility gaps, such as a person entering an office, and later re-appearing, or a person going around the corner into the open world and re-appearing. Due to the nature of the problem, we were forced to employ cues that we knew would persist over that gap in time.

For example, the possession based rules state that identity can be verified on the basis of a person possessing something that only he can possess. So if it were known that a vehicle belonged to an individual and later another individual was

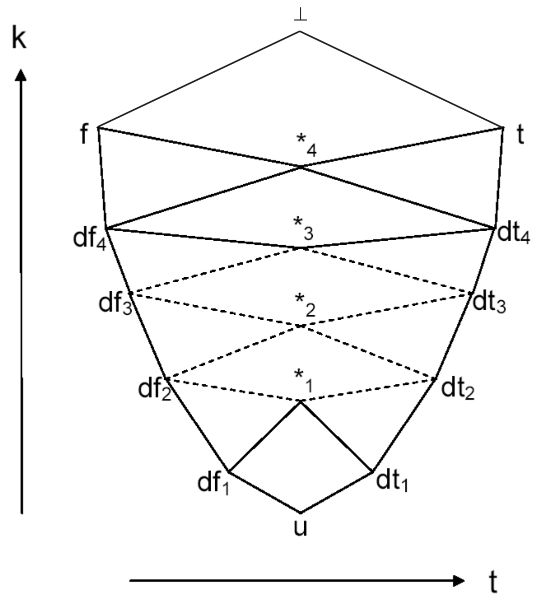


Figure 5.3: Bilattice employed in proposed system.

observed entering that vehicle using a key that he possessed, it was concluded that the two individuals were equal. We were unable to conclude identity, however, based on less persistent objects like bags. For example, if an individual was observed to drop a bag in the scene and disappear from view and after a prolonged period of time, another individual was observed to appear in the scene and pick up the bag, it would not make sense to conclude that the two individuals were equal. The second individual could, for example, be committing a theft. However since here we are concerned with identity maintenance across occlusions which are relatively short visibility gaps, we can employ objects like bags to help in establishing identity. This requires that we make the assumption that bags do not change possession during the visibility gap. In addition to these four categories of rules, we also employ equality axioms of reflexivity, transitivity, and symmetry.

5.4.2 Identity Rules

It should be noted that any rule based on the cues listed above can almost never be definitive - most of them will be default rules. Also, different cues provide us with different amounts of information as they deal with varying degrees of uncertainty. Identity rules are formulated in the MVDL framework with 4 levels of priorities for defaults. Propositions can therefore assume values taken from the set $B = \{u, dt_1, df_1, *_1, dt_2, df_2, *_2, dt_3, df_3, *_3, dt_4, df_4, *_4, t, f\}$. the bilattice for these set of truth values is shown in figure 5.3. The links shown in dotted lines indicate truth values for proposition derived from top-down active search of video. We assume that the definite rules (rules to which we assign truth values t and f) are always correct and therefore there can never be a contradiction between such rules. This assumption results in us ignoring the truth value $\perp$. Following we provide English descriptions of rules at each priority level.

Definite Rules: Definite rules are rules to which we assign truth value of either t or f, (i.e. we have the most confidence in the outcome of these rules). These rules capture knowledge that is always correct and that cannot be proven wrong (while most rules are default rules, definite rules act as stopping rules that terminate the revision of a proposition's belief state).

It is very hard to state that two individuals are definitely equal based on visual observation alone. Irrespective of how much information one includes in such rules, it is always possible to find ways to defeat them. Therefore, in our system we do not have a single rule that definitely infers equality. However, it is possible to state that two individuals are not equal. We do that when we observe them as two distinct individuals at the same instant of time. We also consider the equality axioms of reflexivity, transitivity and symmetry to be definite in nature.

Priority Level 4: Priority level 4 rules are those that compare only two individuals bound by either a closed world event or an identifying object. Occlusions are regarded as closed world events and therefore, an individual going out of view behind an occlusion is expected to reappear. We use this to formulate the following default rule: if we observe an individual enter an occlusion that we believe to be empty (no other individual is currently occluded there) and at a subsequent time, exiting it such that no other individual is observed to enter or exit that occlusion in the time between, then the two individuals are identical. If equality between two individuals is inferred using this rule, it will continue to hold as long as the system has no reason to believe that the occluded region was not empty during the period in question. If at anytime an individual that is not accounted for emerges from the occlusion, all identity assertions made until that point in time based on this rule are suspect and have to be retracted.

Other rules in this category are rules that state that if we observe an individual enter a occlusion and if, while we believe he is still behind, we observe another individual elsewhere in the scene, then these two individuals cannot be equal to each other. Possession based rules also fall in this priority level. An individual can be said to be equal to another individual across an occlusion event if both of them are observed to carry a similar appearing object.

Priority Level 3: Priority level 3 rules are basically the possession based rules mentioned above. The only difference is that if identity is established based on actively searching for possession of a bag by way of top-down feedback (both searching and tracking back in time), we assign to it a truth value of priority level 3. The reason for placing feedback based possession rules one level below pure possession rules is because searching for similar looking objects attached to images of individuals is a less certain process. Our confidence in establishing identity based on what we *think* is a bag being carried by an individual is lower compared to knowing for certain that a bag was indeed carried. These rules are invoked only to resolve belief states of u, $*_1$ or $*_2$.

Priority Levels 2: Appearance matching based on color path length is assigned level 2. Appearance matching rules in general state that if two individuals appear similar, then they must be equal, while if the do not appear similar, then they must be not equal. Color path length based rules, while more accurate in matching appearance, are computationally expensive and are invoked by the high level module only when it is not possible to distinguish individuals based on the simple color histogram based appearance matching i.e. they are only used to resolve $*_1$ or u belief states.

Priority Level 1: Rules based on color histogram based appearance matching are assigned priority level 1 as these have the least information.

5.5 Results

Our system has been implemented as a multi-threaded, C++ application. A Prolog reasoning engine has been embedded within this C++ application. Multivalued default reasoning is implemented using meta-predicates provided by Prolog. The application consists of two kinds of threads: the camera thread (the low level module), which take input from the camera and detects “atomic” events (like entering a door or picking up a bag) and a reasoning thread (high level module), responsible for the high level multivalued default reasoning. The camera thread first performs background subtraction and local tracking. It then detects “atomic” events and syntactically structures them as Prolog facts. The reasoning thread, when first created, starts the Prolog engine and initializes it by inserting into its knowledge base all the predefined rules from the default theory. The reasoning thread is subsequently evoked every few seconds. Every time it runs, it assimilates Prolog facts generated by the camera thread and inserts them into the Prolog engine's knowledge base. Also, for every human observed in the video, it reasons about their identity by applying all applicable equality rules. If it detects that any identity statement is in the “unknown” or any of the “undecided” states, it attempts to actively seek information to emerge from that state. All of the feedback controlled modules are run in a separate thread so they do not disturb the normal bottom-up functioning of the system.

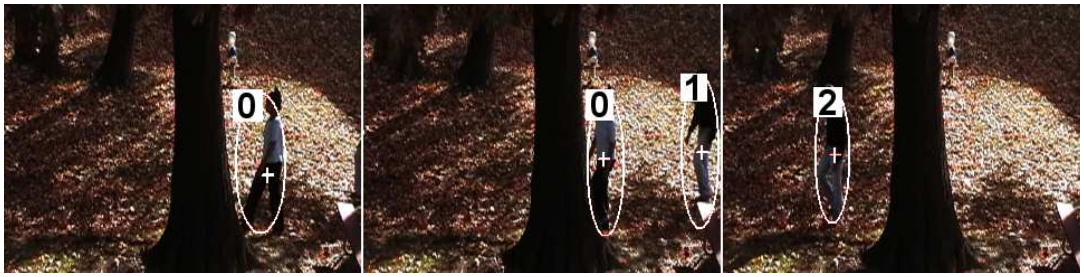


Figure 5.4: Sequence of images showing individual 0 and 1 disappearing from view and 2 subsequently appearing from behind an occlusion

5.5.1 Scenarios

We describe the scenarios used to test the system.

Scenario 7 (See Figure 5.4)**.** *Two individuals 0 and 1 walk behind an occlusion. Individual 0 is wearing a blue shirt and black pants while individual 1 is wearing a black shirt and a blue pants. Individual 2 appears from the occlusion subsequently.*

In scenario 7, the overall color distribution between the two individuals 0 and 1 is similar and therefore, the level 1 rules, based on the color histogram based appearance matching, compute $\phi[equal(2,0)] = dt_1$ and $\phi[equal(2,1)] = dt_1$. However, the system is also able to prove that $\phi[equal(0,1)] = f$ and therefore by transitivity is forced to assign $\phi[equal(2,0)] = *_1$ and $\phi[equal(2,1)] = *_1$. Since the belief states of the identity statements is $*_1$, the high level module directs the system to use the level 2 appearance matching algorithm which employs color as well as spatial distribution of the pixels to match appearances. With this information, the system is now able to correctly conclude $\phi[equal(2,1)] = dt_2$ and $\phi[equal(2,0)] = df_2$.

Scenario 8 (See Figure 5.5)**.** *Two individuals 0 and 1 approach each other and their*

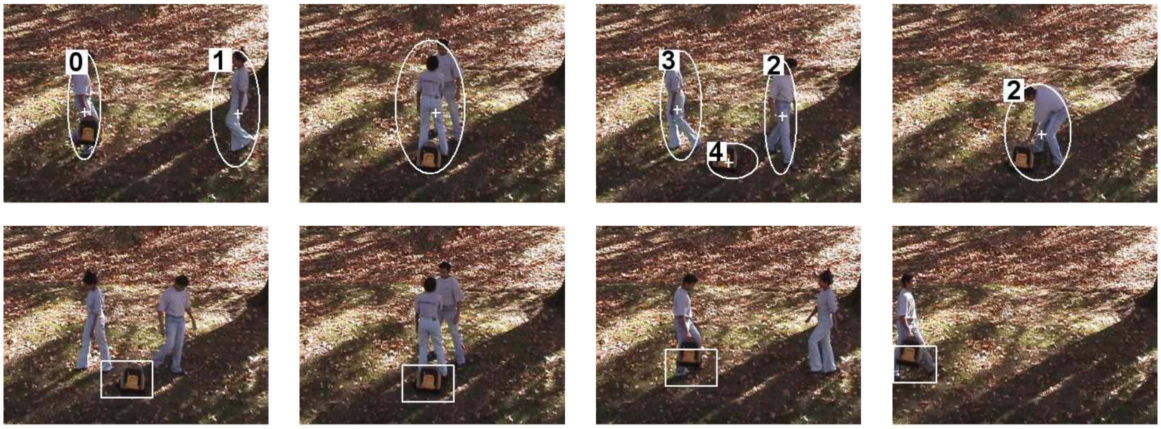


Figure 5.5: Figure described in scenario 2. Top row: Events detected bottom-up. Bottom row: Tracking back of bag

views merge. Subsequently the individuals separate out and are now labelled 2 and 3 by the system. At this point, bag 4 is detected on the ground where 0 and 1 had merged. 3 exits the scene while 2 picks up bag 4 and exits the scene

In this scenario too as in scenario 7, the system concludes $\phi[equal(3,0)] = *_1$, $\phi[equal(3,1)] = *_1$, $\phi[equal(2,0)] = *_1$ and $\phi[equal(2,1)] = *_1$. Application of level 2 rules does not help in this case, as both individuals are dressed alike and the system concludes $\phi[equal(3,0)] = *_2$,$\phi[equal(3,1)] = *_2$, $\phi[equal(2,0)] = *_2$ and $\phi[equal(2,1)] = *_2$. However when individual 2 picks up bag 4, the high level module can now potentially apply possession based level 3 set of rules. Therefore, it directs the low level module to track the bag backward in time from the point when it was first detected. The bag is correctly tracked back to individual 0 and the system concludes $\phi[equal(2,0)] = dt_3$.

Scenario 9 (See Figure 5.1)**.** *Two similar looking individuals 0 and 1 disappear behind an occlusion and are completely lost sight of. Subsequently, individual 3* *appears from the occlusion and drops bag 4 on the ground.*

As in the previous scenario, in this scenario too, since individuals 0 and 1 appear similar, both appearance matching cues are unable to assign to the identity statements any belief state greater than $*_2$. However, at the instant the bag 4 is detected, the high level module sends a search request to the low level module. The object to be searched for is the bag 4 and the objects to be searched within are images of 0 and 1. The system correctly identifies 1 as carrying the bag and revises $\phi[equal(3, 1)]$ from $*_2$ to dt_3.

5.6 Discussions and Summary

This chapter describes the use of context driven top-down image analysis for establishing identity of individuals across short visibility gaps. Use of the MVDL framework allows the system to use the degree of information of various cues to not only combine them in an information theoretic manner but to also detect situations where more information is needed and thus to drive the low level modules

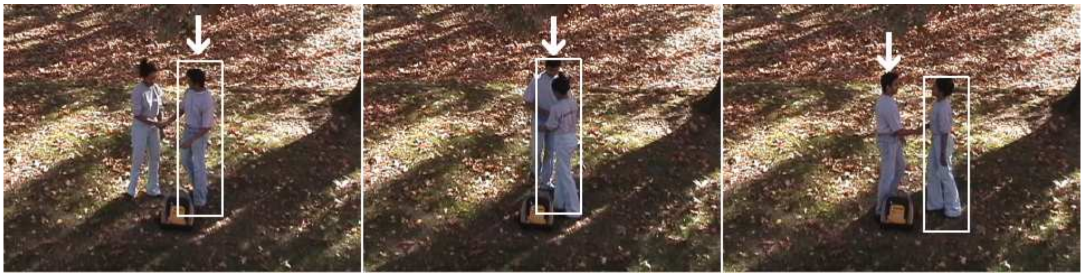


Figure 5.6: Typical tracking failure with conventional mean shift tracker. White arrow (manually inserted) shows correct track

and actively seek information. Maintaining identity of individuals across occlusions is important for any surveillance application. Any system that employs conventional tracking algorithms will fail to handle situations such as those described in the scenarios in Section 5.5 (also see Figure 5.6). By giving the system the capacity to make identity decisions based on any available context, we give it the ability to handle some of these difficult cases. Understandably, not all occlusion events will be successfully handled by our system and most identity decisions will remain in "unknown" or "undecided" states. However, the fact that there exists a deficit of information will be explicitly known. This opens up possibilities for constructing more complex control feedback algorithms that can be used to extract more information from archival video that will help disambiguate.

Chapter 6

Bilattice based Logical Reasoning for Human Detection

6.1 Introduction

The primary objective of an automated visual surveillance system is to observe and understand human behavior and report unusual or potentially dangerous activities/events in a timely manner. Realization of this objective requires at its most basic level the capacity to robustly detect humans from input video. Human detection, however, is a difficult problem. This difficulty arises due to wide variability in appearance of clothing, articulation, view point changes, illumination conditions, shadows and reflections, among other factors. While detectors can be trained to handle some of these variations and detect humans individually as a whole, their performance degrades when humans are only partially visible due to occlusion, either by static structures in the scene or by other humans. Part based detectors are better suited to handle such situations because they can be used to detect the un-occluded parts. However, the process of going from a set of partial body part detections to a set of scene consistent, context sensitive, human hypotheses is far from trivial.

Since part based detectors only learn part of the information from the whole human body, they are typically less reliable and tend to generate large numbers of false positives. Occlusions and local image noise characteristics also lead to missed

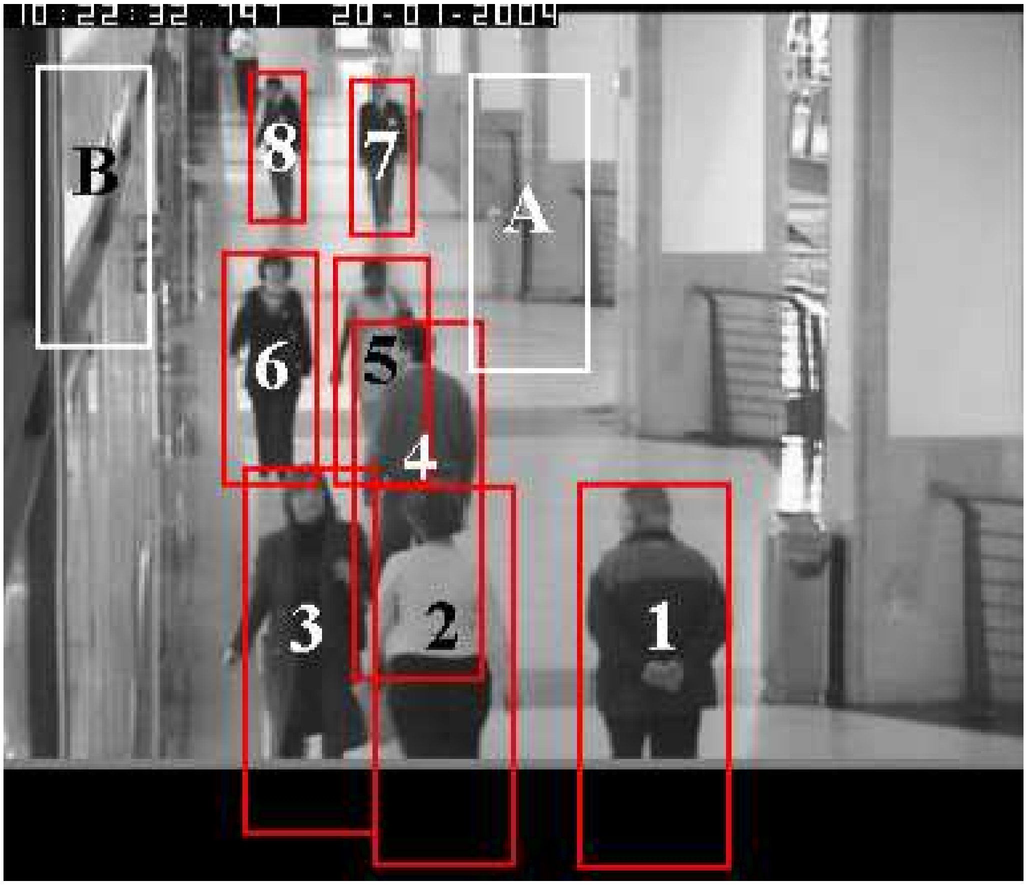


Figure 6.1: Figure showing valid human detections and a few false positives.

detections. It is therefore important to not only exploit contextual, scene geometry and human body constraints to weed out false positives, but also be able to explain as many valid missing body parts as possible to correctly detect occluded humans.

Figure 6.1 shows a number of humans that are occluded by the scene boundary as well as by each other. Ideally, a human detection system should be able to reason about whether a hypothesis is a human or not by aggregating information provided by different sources, both visual and non-visual. For example, in figure 6.1, the system should reason that it is likely that individual 1 is human because two independent sources, the head detector and the torso detector report that it is a human. The absence of legs indicates it is possibly not a human, however this absence can be justified due to their occlusion by the image boundary. Furthermore, hypothesis 1 is consistent with the scene geometry and lies on the ground plane. Since the evidence for it being human exceeds evidence against, the system should decide that it is indeed a human. Similar reasoning applies to individual 4, only its legs are occluded by human 2. Evidence against A and B (inconsistent with scene geometry and not on the ground plane respectively) exceeds evidence in favor of them being human and therefore A and B should be rejected as being valid hypotheses.

This chapter proposes a logic based approach that reasons and detects humans in the manner outlined above. In this framework, knowledge about contextual cues, scene geometry and human body constraints is encoded in the form of rules in a logic programming language and applied to the output of low level parts based detectors. Positive and negative information from different rules, as well as uncertainties from detections are integrated within the bilattice framework. This framework also generates proofs or justifications for each hypothesis it proposes. These justifications (or lack thereof) are further employed by the system to explain and validate, or reject potential hypotheses. This allows the system to explicitly reason about complex interactions between humans and handle occlusions. These proofs are also available to the end user as an explanation of why the system thinks a particular hypothesis is actually a human. We employ a boosted cascade of gradient histograms based detector to detect individual body parts.

We have applied this framework to analyze the presence of humans in static images and have evaluated it on the 'USC pedestrian set B' [93], USC's subset of the CAVIAR dataset [1], that includes images of partially occluded humans (This dataset will henceforth be referred to in this chapter as the USC-CAVIAR dataset). We have also evaluated it on a dataset we collected on our own. In this chapter, we refer to this dataset as Dataset-A.

6.2 Reasoning Framework

To perform the kind of reasoning outlined in section 6.1, one has to specify rules that allow the system to take visual input from the low level detectors and explicitly infer whether or not there exists a human at a particular location. For instance, if we were to employ a head, torso and legs detector, then a possible rule would be:

$$\begin{aligned} human(X,Y,S) \quad \longleftarrow \quad & head(X_h,Y_h,S_h), \\ & torso(X_t,Y_t,S_t), \\ & legs(X_l,Y_l,S_l), \\ & geometry\_constraint(X_h,Y_h,S_h,X_t,Y_t,S_t,X_l,Y_l,S_l), \\ & compute\_center(X_h,Y_h,S_h,X_t,Y_t,S_t,X_l,Y_l,S_l,X,Y,S). \end{aligned}$$

This rule captures the information that if the head, torso and legs detectors were to independently report a detection at some location and scale (by asserting facts $head(X_h,Y_h,S_h)$, $torso(X_t,Y_t,S_t)$, $legs(X_l,Y_l,S_l)$ respectively), and these coordinates respected certain geometric constraints, then one could conclude that there exists a human at that location and scale. A logic programming system would search the input facts to find all combinations that satisfy the rule and report the presence of humans at those locations. Note that this rule will only detect humans that are visible in their entirety. Similar rules can be specified for situations when one or more of the detections are missing due to occlusions or other reasons. There are, however, some problems with a system built on such rule specifications:

1. Traditional logics treat such rules as binary and definite, meaning that every time the body of the rule is true, the head will have to be true. There is no way of saying for example that if the body is true, then only in some cases, the head of the rule holds. In other words, we need to be able to assign some uncertainty values to the rules that captures its reliability.

2. Traditional logics treat facts as binary. We would like to take as input, along with the detection, the uncertainty of the detection also and integrate it into the reasoning framework

3. Traditional logic programming does not have support for explicit negation in the head. There is no easy way of specifying a rule like:

$$\neg human(X, Y, S) \leftarrow \neg scene\_consistent(X, Y, S).$$

and integrating it with positive evidence. Such a rule says that a hypothesis is not a human if it is not consistent with the geometry of the scene.

4. Such a system will not be scalable. We would have to specify one rule for every situation we foresee. If we would like to include in our reasoning the output from another detector, say a hair detector to detect the presence of hair and consequently a head, we would have to re-engineer all our rules to account for

new situations. We would like a framework that allows us to directly include new information without re-engineering.

5. Finally, since we would like our reasoning framework to allow us to specify multiple rules, each contributing some positive or negative information about a particular hypothesis, we need a way of combining these sources of information into a single answer. Traditional logic programming does not have support for such integration of multiple evidence.

6.2.1 Bilattice Theory

Bilattices are algebraic structures introduced by Ginsberg [35] as a uniform framework within which a number of diverse applications in artificial intelligence can be modelled. In [3], it was pointed out that bilattices serve as a foundation of many areas such as logic programming, computational linguistics, distributed knowledge processing, reasoning with imprecise information and fuzzy set theory. In our application, the automatic human detection system is looked upon as a passive rational agent capable of reasoning under uncertainty. Uncertainties assigned to the rules that guide reasoning, as well as detection uncertainties reported by the low level detectors, are taken from a set structured as a bilattice. These uncertainty measures are ordered along two axes, one along the source's[1] degree of information and the other along the agent's degree of belief. As we will see, this structure

[1] A single rule applied to a set of facts is referred to as a source here. There can be multiple rules deriving the same proposition (both positive and negative forms of it) and therefore we have multiple sources of information.

allows us to address all of the issues raised in the previous section and provides a uniform framework which not only permits us to encode multiple rules for the same proposition, but also allows inference in the presence of contradictory information from different sources.

Definition 11 (Lattice). *A lattice is a set L equipped with a partial ordering $\leq$ over its elements, a greatest lower bound (glb) and a lowest upper bound (lub) and is denoted as $\mathcal{L} = (L, \leq)$ where glb and lub are operations from $L \times L \rightarrow L$ that are idempotent, commutative and associative. Such a lattice is said to be* ***complete****, iff for every finite nonempty subset M of L, there exists a unique lub and glb.*

Definition 12 (Bilattice [35]). *A bilattice is a triple $\mathcal{B} = (B, \leq_t, \leq_k)$, where B is a nonempty set containing at least two elements and $(B, \leq_t)$, $(B, \leq_k)$ are complete lattices.*

Informally a bilattice is a set, B, of uncertainty measures composed of two complete lattices $(B, \leq_t)$ and $(B, \leq_k)$ each of which is associated with a partial order $\leq_t$ and $\leq_k$ respectively. The $\leq_t$ partial order (agent's degree of belief) indicates how true or false a particular value is, with f being the minimal and t being the maximal while the $\leq_k$ partial order indicates how much is known about a particular proposition. The minimal element here is $\perp$ (completely unknown) while the maximal element is $\top$ (representing a contradictory state of knowledge where a proposition is both true and false). The glb and the lub operators on the $\leq_t$ partial order are $\wedge$ and $\vee$ and correspond to the usual logical notions of conjunction and distinction, respectively. The glb and the lub operators on the $\leq_k$ partial order are

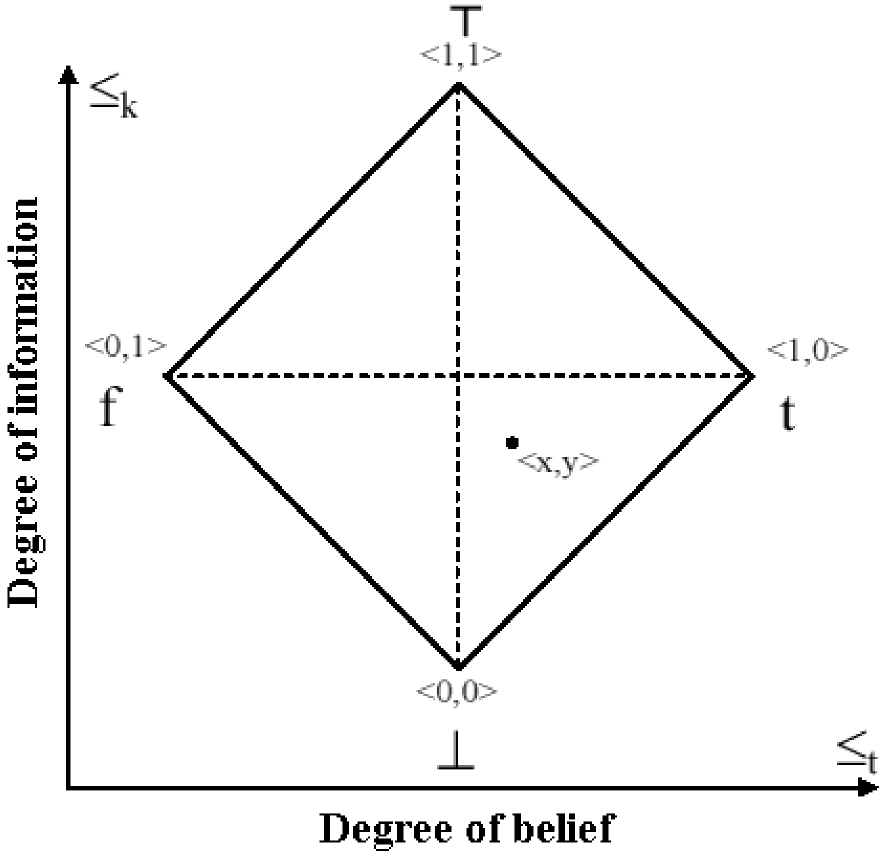


Figure 6.2: The bilattice square $([0,1]^2, \leq_t, \leq_k)$

$\otimes$ and $\oplus$, respectively, where $\oplus$ corresponds to the combination of evidence from different sources or lines of reasoning while $\otimes$ corresponds to the consensus operator. A bilattice is also equipped with a negation operator $\neg$ that inverts the sense of the $\leq_t$ partial order while leaving the $\leq_k$ partial order intact and a conflation operator $-$ which inverts the sense of the $\leq_k$ partial order while leaving the $\leq_t$ partial order intact.

The intuition is that every piece of knowledge, be it a rule or an observation from the real world, provides different degrees of information. An agent that has to reason about the state of the world based on this input, will have to translate the source's degree of information, to its own degree of belief. Ideally, the more information a source provides, the more strongly an agent is likely to believe it (i.e closer to the extremities of the t-axis) . The only exception to this rule being the case of contradictory information. When two sources contradict each other, it will cause the agent's degree of belief to decrease despite the increase in information content. It is this decoupling of the sources and the ability of the agent to reason independently along the truth axis that helps us address the issues raised in the previous section. It is important to note that the line joining $\perp$ and $\top$ represents the line of indifference. If the final uncertainty value associated with a hypothesis lies along this line, it means that the degree of belief for and degree of belief against it cancel each other out and the agent cannot say whether the hypothesis is true or false. Ideally the final uncertainty values should be either $\langle 0, 1\rangle$ or $\langle 1, 0\rangle$, but noise in observation as well as less than completely reliable rules ensure that this is almost never the case. The horizontal line joining t and f is the line of consistency. For any point along this line, the degree of belief for will be exactly equal to 1-degree of belief against and thus the final answer will be exactly consistent.

Definition 13 (Rectangular Bilattice [69])**.** *Let $\mathcal{L} = (L, \leq_L)$ and $\mathcal{R} = (R, \leq_R)$ be two complete lattices. A rectangular bilattice is a structure $\mathcal{L} \odot \mathcal{R} = (L \times R, \leq_t, \leq_k)$, where for every $x_1, y_1 \in \mathcal{L}$ and $x_2, y_2 \in \mathcal{R}$,*

1. *$\langle x_1, x_2\rangle \leq_t \langle y_1, y_2\rangle \Leftrightarrow x_1 \leq_L y_1$ and $x_2 \geq_R y_2$,*

2. *$\langle x_1, x_2\rangle \leq_k \langle y_1, y_2\rangle \Leftrightarrow x_1 \leq_L y_1$ and $x_2 \leq_R y_2$,*

An element $\langle x_1, x_2\rangle$ of the rectangular bilattice $\mathcal{L} \odot \mathcal{R}$ may be interpreted such that x_1 represents the amount of belief for some assertion while x_2 represents the amount of belief against it. If we denote the glb and lub operations of complete lattices $\mathcal{L} = (L, \leq_L)$, and $\mathcal{R} = (R, \leq_R)$ by $\wedge_L$ and $\vee_L$, and $\wedge_R$ and $\vee_R$ respectively, we can define the glb and lub operations along each axis of the bilattice $\mathcal{L} \odot \mathcal{R}$ as follows:

$$\begin{aligned}
\langle x_1, x_2 \rangle \wedge \langle y_1, y_2 \rangle &= \langle x_1 \wedge_L y_1, x_2 \vee_R y_2 \rangle, \\
\langle x_1, x_2 \rangle \vee \langle y_1, y_2 \rangle &= \langle x_1 \vee_L y_1, x_2 \wedge_R y_2 \rangle, \\
\langle x_1, x_2 \rangle \otimes \langle y_1, y_2 \rangle &= \langle x_1 \wedge_L y_1, x_2 \wedge_R y_2 \rangle, \\
\langle x_1, x_2 \rangle \oplus \langle y_1, y_2 \rangle &= \langle x_1 \vee_L y_1, x_2 \vee_R y_2 \rangle, \qquad (6.1)
\end{aligned}$$

Of interest to us in our application is a particular class of rectangular bilattices where $\mathcal{L}$ and $\mathcal{R}$ coincide. These structures are called *squares* and $\mathcal{L} \odot \mathcal{L}$ is abbreviated as $\mathcal{L}^2$. Since detection likelihoods reported by the low level detectors are typically normalized to lie in the [0,1] interval, the underlying lattice that we are interested in is $\mathcal{L} = ([0, 1], \leq)$. The bilattice that is formed by $\mathcal{L}^2$ is depicted in figure 6.2. Each element in this bilattice is a tuple with the first element encoding evidence for a proposition and the second encoding evidence against. In this bilattice, the element f (false) is denoted by the element $\langle 0, 1 \rangle$ indicating, no evidence for but full evidence against, similarly element t is denoted by $\langle 1, 0 \rangle$, element $\perp$ by $\langle 0, 0 \rangle$ indicating no information at all and $\top$ is denoted by $\langle 1, 1 \rangle$. To fully define glb and lub operators along both the axes of the bilattice as listed in equations 6.1, we need to define the glb and lub operators for the lattice $([0, 1], \leq)$. A popular choice for such operators is typically triangular-norms and triangular-conorms. Triangular norms and conorms were introduced by Schweizer and Sklar [77] to model the distances in probabilistic metric spaces. Triangular norms are used to model the glb operator and the triangular conorm to model the lub operator within each lattice.

Definition 14 (triangular norm)**.** *A mapping*

$\mathcal{T} : [0,1] \times [0,1] \rightarrow [0,1]$

is a triangular norm (t-norm) iff it is symmetric, associative, non-decreasing in each argument and $\mathcal{T}(a,1) = a, \forall a \in [0,1]$. *In other words, any t-norm* $\mathcal{T}$ *satisfies the properties:*

- Symmetry: $\mathcal{T}(x,y) = \mathcal{T}(y,x), \forall x,y \in [0,1]$

- Associativity: $\mathcal{T}(x,\mathcal{T}(y,z)) = \mathcal{T}(\mathcal{T}(x,y),z), \forall x,y,z \in [0,1]$.

- Monotonicity:$\mathcal{T}(x,y) \leq \mathcal{T}(x',y') if x \leq x' and\ y \leq y'$

- One identity: $\mathcal{T}(x,1) = x, \forall x \in [0,1]$.

Definition 15 (triangular conorm)**.** *A mapping*

$\mathcal{S} : [0,1] \times [0,1] \rightarrow [0,1]$

is a triangular conorm (t-conorm) iff it is symmetric, associative, non-decreasing in each argument and $\mathcal{S}(a,0) = a, \forall a \in [0,1]$. *In other words, any t-conorm* $\mathcal{S}$ *satisfies the properties:*

- Symmetry: $\mathcal{S}(x,y) = \mathcal{S}(y,x), \forall x,y \in [0,1]$

- Associativity: $\mathcal{S}(x,\mathcal{S}(y,z)) = \mathcal{S}(\mathcal{S}(x,y),z), \forall x,y,z \in [0,1]$.

- Monotonicity:$\mathcal{S}(x,y) \leq \mathcal{S}(x',y') if x \leq x' and\ y \leq y'$

- Zero identity: $\mathcal{S}(x,0) = x, \forall x \in [0,1]$.

if $\mathcal{T}$ is a t-norm, then the equality $\mathcal{S}(a,b) = 1 - \mathcal{T}(1-a, 1-b)$ defines a t-conorm and we say $\mathcal{S}$ is derived from $\mathcal{T}$. There are number of possible t-norms and t-conorms one can choose. In our application, for the underlying lattice, $\mathcal{L} = ([0,1], \leq)$, we choose the t-norm such that $\mathcal{T}(x,y) \equiv x \wedge_L y = xy$ and consequently choose the t-conorm as $\mathcal{S}(x,y) \equiv x \vee_L y = x + y - xy$. Based on this, the glb and lub operators for each axis of the bilattice B can then be defined as per equation 6.1.

Assume the following set of rules and facts:

Rules	**Facts**
$\phi(human(X,Y,S) \leftarrow head(X,Y,S)) = \langle 0.40, 0.60\rangle$	$\phi(head(25,95,0.9)) = \langle 0.90, 0.10\rangle$
$\phi(human(X,Y,S) \leftarrow torso(X,Y,S)) = \langle 0.30, 0.70\rangle$	$\phi(torso(25,95,0.9)) = \langle 0.70, 0.30\rangle$
$\phi(\neg human(X,Y,S) \leftarrow \neg scene\_consistent(X,Y,S)) = \langle 0.90, 0.10\rangle$	$\phi(\neg scene\_consistent(25,95,0.9)) = \langle 0.80, 0.20\rangle$

Inference is performed as follows:

$$cl(\phi)(human(25,95,0.9)){=}\langle 0,0\rangle \vee \Big[\langle 0.4,0.6\rangle \wedge \langle 0.9,0.1\rangle\Big] \oplus \langle 0,0\rangle \vee \Big[\langle 0.3,0.7\rangle \wedge \langle 0.7,0.3\rangle\Big] \oplus \neg\Bigg(\langle 0,0\rangle \vee \Big[\langle 0.9,0.1\rangle \wedge \langle 0.8,0.2\rangle\Big]\Bigg)$$

$$=\langle 0.36,0\rangle \oplus \langle 0.21,0\rangle \oplus \neg\langle 0.72,0\rangle = \langle 0.4944,0\rangle \oplus \langle 0,0.72\rangle = \langle 0.4944,0.72\rangle$$

Figure 6.3: Example showing inference using closure within a $([0,1]^2, \leq_t, \leq_k)$ bilattice

6.2.2 Inference

Inference in bilattice based reasoning frameworks is performed by computing the closure over the truth assignment.

Definition 16 (Truth Assignment)**.** *Given a declarative language L, a truth assignment is a function $\phi : L \rightarrow B$ where B is a bilattice on truth values or uncertainty measures.*

Definition 17 (Closure)**.** *Let $\mathcal{K}$ be the knowledge base and ϕ be a truth assignment, labelling each every formula $k \in \mathcal{K}$, then the closure over ϕ, denoted $cl(\phi)$ is the truth assignment that labels information entailed by $\mathcal{K}$.*

For example, if ϕ labels sentences $\{p, q \leftarrow p\} \in \mathcal{K}$ as $\langle 1, 0 \rangle$ (true); i.e. $\phi(p) = \langle 1, 0 \rangle$ and $\phi(q \leftarrow p) = \langle 1, 0 \rangle$, then $cl(\phi)$ should also label q as $\langle 1, 0 \rangle$ as it is information entailed by $\mathcal{K}$. Entailment is denoted by the symbol '$\models$' ($\mathcal{K} \models q$).

If $S \subset L$ is a set of sentences entailing q, then the uncertainty measure to be assigned to the conjunction of elements of S is

$$\bigwedge_{p \in S} cl(\phi)(p) \tag{6.2}$$

This term represents the conjunction of the closure of the elements of S[2]. It is important to note that this term is merely a contribution to the final uncertainty measure of q and not the final uncertainty measure itself. The reason it is merely a contribution is because there could be other sets of sentences S that entail q

[2] Recall that $\wedge$ and $\vee$ are glb and lub operators along the $\leq_t$ ordering and $\otimes$ and $\oplus$ along $\leq_k$ axis. $\bigwedge, \bigvee, \bigotimes, \bigoplus$ are their infinitary counterparts such that $\bigoplus_{p \in S} p = p_1 \oplus p_2 \oplus \cdots$ and so on

representing different lines of reasoning (or, in our case, different rules). The contributions of these sets of sentences need to be combined using the $\oplus$ operator along the information ($\leq_k$) axis. Also, if the expression in 6.2 evaluates to false, then its contribution to the value of q should be $\langle 0,0\rangle$ (unknown) and not $\langle 0,1\rangle$ (false). These arguments suggest that the closure over ϕ of q is

$$cl(\phi)(q) = \bigoplus_{S \models q} \perp \vee [\bigwedge_{p \in S} cl(\phi)(p)] \tag{6.3}$$

where $\perp$ is $\langle 0,0\rangle$. This is however, only part of the information. We also need to take into account the set of sentences entailing $\neg q$ and aggregating this information yields the following expression

$$cl(\phi)(q) = \bigoplus_{S \models q} \perp \vee [\bigwedge_{p \in S} cl(\phi)(p)] \oplus \neg \bigoplus_{S \models \neg q} \perp \vee [\bigwedge_{p \in S} cl(\phi)(p)] \tag{6.4}$$

For more details see [35]

Figure 7.1 shows an example illustrating the process of computing the closure as defined above by combining evidence from three sources. In this example, the final uncertainty value computed is $\langle 0.4944, 0.72\rangle$. This indicates that evidence against the hypothesis at (25,95) at scale 0.9 exceeds evidence in favor of and, depending on the final threshold for detection, this hypothesis is likely to be rejected.

6.2.3 Negation

Systems such as this typically employ different kinds of negation. One kind of negation that has already been mentioned earlier is $\neg$. This negation flips the bilattice along the $\leq_t$ axis while leaving the ordering along the $\leq_k$ axis unchanged. Another important kind of negation is negation by failure to prove, denoted by *not*. $not(A)$ succeeds if A fails. This operator flips the bilattice along both the $\leq_t$ axis as well as the $\leq_k$ axis. Recall that $-$ was defined as the conflation operator in section 6.2. Therefore, $\phi(not(A)) = \neg - \phi(A)$. In other words, if A evaluates to $\langle 0, 0\rangle$, then $not(A)$ will evaluate to $\langle 1, 1\rangle$. This operator is important when we want to detect the absence of a particular body part for a hypothesis.

6.3 Detection System

Rules can now be defined within this bilattice framework to handle complex situations, such as humans being partially occluded by static structures in the scene or by other humans. Each time one of the detectors detects a body part, it asserts a logical fact of the form $\phi(head(x, y, s)) = \langle\alpha, \beta\rangle$, where α is the measurement score the detector returns at that location and scale in the image and, for simple detectors, β is $1 - \alpha$. Rules are specified similarly as $\phi(human(X, Y, S) \leftarrow \cdots) = \langle\gamma, \delta\rangle$. γ and δ are learnt as outlined in subsection 7.3.2. We start by initializing a number of initial hypotheses based on the low level detections. For example, if the head detector detects a head and asserts fact $\phi(head(75, 225, 1.25)) = \langle 0.95, 0.05\rangle$[3], the system records that there exists a possible hypothesis at location (75,225) at scale 1.25 and submits the query $human(75, 225, 1.25)$ to the logic program where support for and against it is gathered and finally combined into a single answer within the

[3] Note that the coordinates here are not the centers of the body parts, but rather the centers of the body

bilattice framework. Projecting the final uncertainty value onto the $\langle 0,1\rangle - \langle 1,0\rangle$ axis, gives us the final degree of belief in the hypothesis. We will now provide English descriptions of some of the rules employed in our system.

6.3.1 Rule Specification

Rules in such systems can be learnt automatically; however, such approaches are typically computationally very expensive. We manually encode the rules while automatically learning the uncertainties associated with them. The rules fall into three categories: Detector based, Geometry based and Explanation based

Detector based: These are the simplest rules that hypothesize that a human is present at a particular location if one or more of the detectors detects a body part there. In other words, if a head is detected at some location, we say there exists a human there. Note that this rule, along with all the others, is not a definite rule and has uncertainty (say $\langle \alpha, \beta \rangle$)associated with it, meaning that if a head is detected, then α is the likelihood that it is indeed a human. There are four such rules, one each for the head,torso, legs and fullbody based detectors.

Geometry based: Geometry based rules validate or reject human hypotheses based on geometric and scene information. This information is entered a priori in the system at setup time. We employ information about expected height of people and regions of expected foot location. The expected image height rule is based on ground plane information and anthropometry. Fixing a gaussian at an adult human's expected physical height allows us to generate scene consistency likelihoods for a particular hypothesis given its location and size. The expected foot location region is a region demarcated in the image outside of which no valid feet can occur and therefore serves to eliminate false positives.

Explanation based: Explanation based rules are the most important rules for a system that has to handle occlusions. The idea here is that if the system does not detect a particular body part, then it must be able to explain its absence for the hypothesis to be considered valid. If it fails to explain a missing body part, then it is construed as evidence against the hypothesis being a human. Absence of body parts is detected using logic programming's 'negation as failure' operator (*not*). $not(A)$ succeeds when A evaluates to $\langle 0, 0 \rangle$ as described in section 6.2.3. A valid explanation for missing body part could either be due to occlusions by static objects or due to occlusions by other humans.

Explaining missed detections due to occlusions by static objects is straightforward. At setup, all static occlusions are marked. Image boundaries are also treated as occlusions and marked as shown in figure 6.1(black area at bottom of figure). For a given hypothesis, the fraction of overlap of the missing body part with the static occlusion is computed and reported as the uncertainty of occlusion. The process is similar for occlusions by other human hypotheses, with the only difference being that, in addition to the degree of occlusion, we also take into account the degree of confidence of the hypothesis that is responsible for the occlusion, as illustrated in the rule below:

$$
\begin{aligned}
human(X,Y,S) \;\leftarrow\;\; & not(torso(X_t,Y_t,S_t),\\
& torso\_body\_consistent(X,Y,S,X_t,Y_t,S_t)),\\
& torso\_occluded(X,Y,S,X_o,Y_o,S_o),\\
& Y_o > Y, human(X_o,Y_o,S_o). \qquad (6.5)
\end{aligned}
$$

This rule will check to see if $human(X,Y,S)$'s torso is occluded by $human(X_o,Y_o,S_o)$ under condition that $Y_o > Y$, meaning the occluded human is behind the 'occluder'[4] There is a similar rule for legs and also rules deriving $\neg human$ in the absence of explanations for missing parts.

6.3.2 Learning

Given a rule of the form $A \leftarrow B_1, B_2, \cdots, B_n$, a confidence value of

$$\left\langle \mathcal{N}(A|B_1,B_2,\cdots,B_n), \mathcal{N}(\neg A|B_1,B_2,\cdots,B_n) \right\rangle$$

is computed, where $\mathcal{N}(A|B_1,B_2,\cdots,B_n)$ is the fraction of times A is true when $B_1,B_2,\cdots,B_n$ is true. It is important to note that the presence of a non-zero value for $\mathcal{N}(\neg A|B_1,B_2,\cdots,B_n)$ does not imply the existence of a rule of the form

[4] The reader might notice that calling the $human(X_o,Y_o,S_o)$ within the definition of a 'human' rule will cause the system to infer the presence of $human(X_o,Y_o,S_o)$ from scratch. This rule has been presented in such a manner merely for ease of explication. In practice, we maintain a table of inferences that the query, $human(X_o,Y_o,S_o)$, can tap into for unification without re-deriving anything. Also we derive everything from the bottom of the image to the top, so $human(X_o,Y_o,S_o)$ is guaranteed to unify.

$$\neg A \leftarrow B_1, B_2, \cdots, B_n.$$

Total:	human(243,253,1.5)	$\langle 0.484055, 0.162474\rangle$
+ve evidence:	head(244.5, 247.5, 1.5)	$\langle 1, 0\rangle$
	torso(243, 253,1.5)	$\langle 1, 0\rangle$
	fullbody(243, 256.5,1.5)	$\langle 0.9371, 0\rangle$
	on_ground_plane(243, 253, 1.5),	$\langle 1, 0\rangle$
	scene_consistent(243, 253, 1.5),	$\langle 0.954835, 0.045165\rangle$
	not((legs(_G3817, _G3818,_G3819),	
	legs_body_consistent(243, 253, 1.5, _G3817,_G3818, _G3819)))	$\langle 1, 1\rangle$
	is_part_occluded(219.0, 253.0, 267.0, 325.0)	$\langle 0.569444, 0.430556\rangle$
-ve evidence:	¬ scene_consistent(243, 253, 1.5)	$\langle 0.045165, 0.954835\rangle$
	not((legs(_G3984,_G3985, _G3986),	
	legs_body_consistent(243, 253, 1.5, _G3984,_G3985, _G3986)))	$\langle 1, 1\rangle$

Table 6.1: Proof for human marked as ‘1’ in figure 6.1

6.3.3 Generating Proofs

As mentioned earlier, in addition to using the explanatory ability of logical rules, we can also provide these explanations to the user as justification of why the system believes that a given hypothesis is a human. The system provides a straightforward technique to generate proofs from its inference tree. Since all of the bilattice based reasoning is encoded as meta logical rules in a logic programming language, it is easy to add predicates that succeed when the rule fires and propagate character strings through the inference tree up to the root where they are aggregated and displayed. Such proofs can either be dumps of the logic program itself or be English text. In our implementation, we output the logic program as the proof tree.

6.4 Body Part Detector

Our human body part detectors are inspired by [100]. Similar to their approach we train a cascade of svm-classifiers on histograms of gradient orientations. Instead of the hard threshold function suggested in their chapter, we apply a sigmoid function to the output of each svm. These softly thresholded functions are combined using a boosting algorithm [25]. After each boosting round, we calibrate the probability of the partial classifier based on evaluation set, and set cascade decision thresholds based on the sequential likelihood ratio test similar to [83]. To train the parts-based detector, we restrict the location of the windows used during the feature computation to the areas corresponding to the different body parts (head/shoulder, torso, legs). The number of layers used in fullbody, head, torso and leg detectors were 12, 20, 20, and 7 respectively.

Total:	human(154,177,1.25)	$\langle 0.359727, 0.103261 \rangle$
+ve evidence:	head(154, 177, 1.25)	$\langle 0.94481, 0 \rangle$
	torso(156.25, 178.75, 1.25)	$\langle 0.97871, 0 \rangle$
	on_ground_plane(154, 177, 1.25)	$\langle 1, 0 \rangle$
	scene_consistent(154, 177, 1.25)	$\langle 0.999339, 0.000661 \rangle$
	not((legs(_G7093,_G7094, _G7095),	
	legs_body_consistent(154, 177, 1.25,_G7093, _G7094, _G7095)))	$\langle 1, 1 \rangle$
	is_part_occluded(134.0, 177.0, 174.0, 237.0)	$\langle 0.260579, 0.739421 \rangle$
-ve evidence:	¬scene_consistent(154, 177, 1.25)	$\langle 0.000661, 0.999339 \rangle$
	not((legs(_G7260, _G7261, _G7262),	
	legs_body_consistent(154, 177, 1.25,_G7260, _G7261, _G7262)))	$\langle 1, 1 \rangle$

Table 6.2: Proof for human marked as ‘4’ in figure 6.1

6.5 Experiments

The framework has been implemented in C++ with an embedded Prolog reasoning engine. The C++ module initializes the Prolog engine by inserting into its knowledge base all predefined rules. Information about scene geometry, and static occlusions is specified through the user interface, converted to logical facts and inserted into the knowledge base. The C++ module then runs the detectors on the given image and structures their output as logical facts for the Prolog knowledge base. Initial hypotheses are created based on these facts and then evidence for or against these hypotheses is searched for by querying for them. We will first describe some qualitative results and show how our system reasons and resolves difficult scenarios, and then describe quantitative results on the USC-CAVIAR dataset as well as on Dataset-A.

6.5.1 Qualitative Results

Tables 6.1 and 6.2 list the proofs for humans 1 and 4 from figure 6.1. In both cases, the head and torso are visible while the legs are missing. In case of human 1, it is due to occlusion by the image boundary (which has been marked as a static occlusion) and in case of human 4 due to occlusion by human 2. In figures 6.1 and 6.2, variables starting with $\_G\cdots$ are non-unified variables in Prolog, meaning that legs cannot be found and therefore the variables of the predicate legs cannot be instantiated. It can be seen that in both cases, evidence in favor of the hypothesis exceeds that against.

6.5.2 Numerical Results

We applied our framework to the set of static images taken from USC-CAVIAR dataset. This dataset, a subset of the original CAVIAR [1] data, contains 54 frames with 271 humans of which 75 humans are partially occluded by other humans and 18 humans are occluded by the scene boundary. This data is not part of our training set. We have trained our parts based detector on the MIT pedestrian dataset [70]. For training purposes, the size of the human was 32x96 centered and embedded within an image of size 64x128. We used 924 positive images and 6384 negative images for training. Figure 6.4 shows the ROC curves for our parts based detectors as well as for the full reasoning system. "Full Reasoning*", in Figure 6.4, is the ROC curve on the 75 occluded humans and table 6.3 lists detection rates for these 75 humans for different degrees of occlusion. ROC curves for part based detectors represent detections that have no prior knowledge about scene geometry or other anthropometric constraints. It can be seen that performing high level reasoning over low level part based detections, especially in presence of occlusions, greatly increases overall performance. We have also compared the performance of our system with the results reported by Wu and Nevatia [93] on the same dataset. We have taken results reported in their original paper and plotted them in figure 6.4 as well as listed them in table 6.3. As can be seen, results from both systems are comparable.

We also applied our framework on another set of images taken from a dataset we collected on our own (in this chapter we refer to it as Dataset-A). This dataset contains 58 images (see figure 6.5) of 166 humans, walking along a corridor, 126 of

Occlusion Degree(%)	>70	70-50	50-25
Human#	10	31	34
Detection Rate(%)	87	91.4	92.6
Detection Rate(%) (Wu Nevatia [93])	80	90.3	91.2

Table 6.3: Detection rates on the USC-CAVIAR dataset for different degrees of occlusion on the 75 humans that are occluded by other humans (with 19 false alarms). Results of [93] on the same dataset are copied from their original paper.

whom are occluded 30% or more, 64 by the image boundary and 62 by each other. Dataset-A is significantly harder than the USC-CAVIAR dataset due to heavier occlusions (44 humans are occluded 70% or more), perspective distortions (causing humans to appear tilted), and due to the fact that many humans appear in profile view. Figure 6.6 shows the ROC curves for this dataset. It can be seen that the low level detectors as well as the full body detector perform worse here than on the USC-CAVIAR data, however, even in such a case, the proposed logical reasoning approach gives a big improvement in performance. If the performance of the low level detectors is further enhanced (to take in account profile views and handle perspective distortions), then results of high level reasoning will further improve. This is part of our future work.

6.6 Discussions and Future Work

We have described a logical reasoning approach for human detection that takes input from multiple sources of information, both visual and non-visual, and integrates them into a single hypothesis within the bilattice framework. Use of logical reasoning permits to explicitly reason about complex interactions between humans as well as with the environment and thus handle occlusions. Structuring of this reasoning within the bilattice framework makes it scalable, so information from new sources can be added easily. The system also generates proofs for validation by the operator. Such a formulation frees us from having to estimate an exponential number of conditional interdependencies between propositions, unlike in statistical frameworks. As can be seen from the closure expression (equation 6.4), complexity of inference in such systems is linear in the number of rules and its constituent propositions. In the future we would like to extend this system to reason explicitly about temporal information thus helping us not only track humans, but also to define models for and recognize human activities within a single framework.

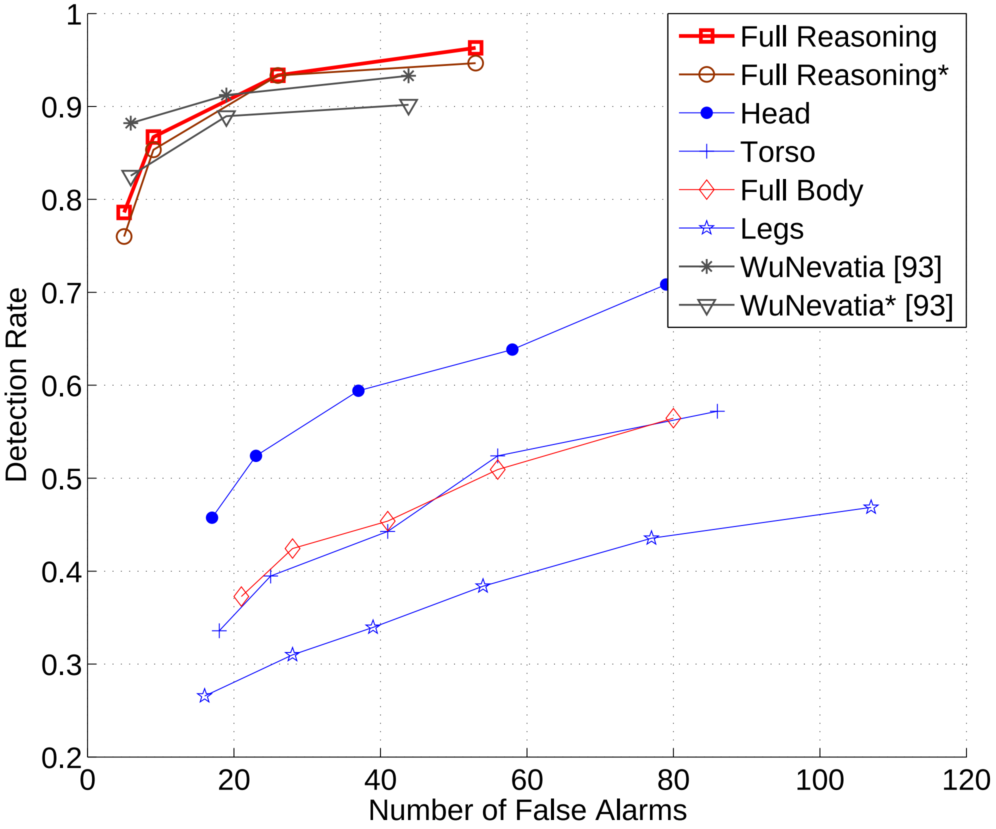


Figure 6.4: ROC curves for evaluation on the USC-CAVIAR dataset. Full Reasoning* is ROC curve for 75 humans occluded by other humans. Results of [93] on the same dataset are copied from their original paper. WuNevatia* is ROC curve for the 75 humans occluded by other humans

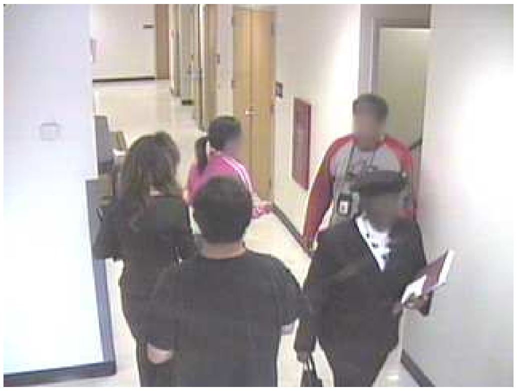


Figure 6.5: An image from Dataset-A

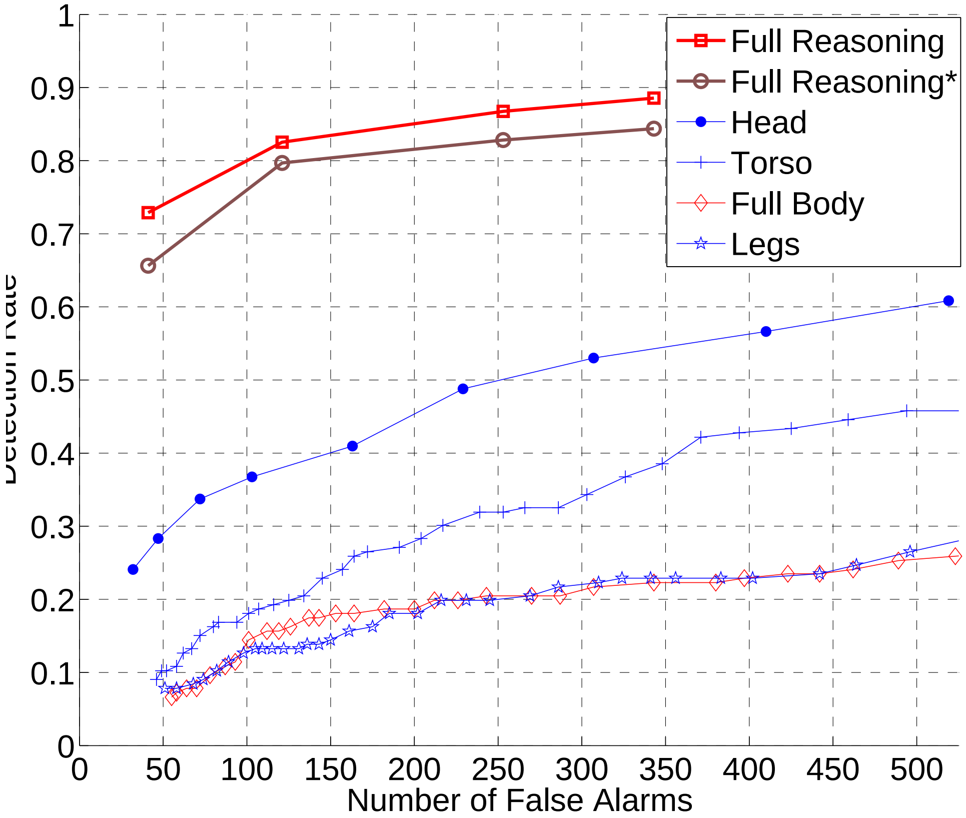


Figure 6.6: ROC curves for evaluation on Dataset-A. Full Reasoning* is ROC curve for 126 occluded humans.

Chapter 7

Bilattice based Logical Reasoning for Preference Modeling

7.1 Introduction

Preference modeling involves analysis and prediction of users' preferences for a set of objects based on their historical preference data and information about other users. Collaborative (social) filtering [79] is an example of preference modeling and prediction that is actively employed on the internet to recommend books, movies, and other commodities. In collaborative filtering applications, there exist, potentially, a large number of cues that can contribute to making a final prediction for a given user. These cues could be correlations between user age, gender, occupation, education, income, etc and properties of the objects such as genre, year of release/publication. Other cues such as individual preference for a particular actor or author might also provide cues for a final user preference. Since many of these cues are typically generalizations based on historical data, they will never be fully indicative of a particular user's preference and often times will be contradictory in nature. Contradictions also exist because of missing data and the fact that users themselves are not always consistent in their ratings.

In this chapter we are primarily concerned with personalized movie recommendation. The task is to come up with an ordered list of movies a particular user, say Cathy, is likely to enjoy. The system first asks Cathy to rate a list of movies (typically on a scale of 1 - 5) that she has previously viewed. Based on her ratings and the ratings provided by other viewers for the same movies the system learns her preferences and returns a ranked/ordered list of recommendations.

7.1.1 Motivation

Ideally, a preference prediction system should be able to reason about the preference of a given user by aggregating information from different sources. For example, the system should reason that Cathy is likely to prefer movie `Casablanca` (Genre:Romance) over `The Terminator` (Genre:Action), because there exists another user, Alice, who prefers `Casablanca` over `The Terminator` and Alice has consistently rated movies similarly to Cathy. The existence of another user Tom, who has also been rating movies similarly to Cathy, but who prefers `The Terminator` over `Casablanca`, should indicate that perhaps Cathy does not prefer `Casablanca` over `The Terminator` after all. However, this anomaly may be explained if Tom is known to be a 17 year old male and data indicates that teenage males typically rate action movies higher than romance. Cathy's preference for `Casablanca` over `The Terminator` should be further strengthened by the knowledge that Cathy has historically rated movies that star Humphrey Bogart 4 stars or higher and has also historically rated romance movies higher.

7.1.2 Overview

We propose a logic based approach that reasons and predicts preferences in the manner outlined above. In this framework, knowledge about inferring preference relations based on data correlations, historical user preferences, and other constraints is encoded as rules in a logic programming language. Historical user preferences and data correlations are stored as logical facts and these facts, in conjunction with the logical rules are employed in making predictions about a given user's preference over a pair of movies. Uncertainties are associated with both the logical rules (encoding the reliability of the rule), as well as to the logical facts (encoding degree of belief in their veracity). Positive and negative information from the rules and facts as well as their associated uncertainties are combined within a bilattice framework. The bilattice formulation, permits encoding of multiple rules for the same proposition while allowing for inference in the presence of contradictory information from different sources. This approach can also generate proofs or justifications for each prediction it makes. These proofs can be made available to the system programmer for debugging purposes as well to users as an indication of why the system thinks they are likely to prefer a particular movie. We have applied the bilattice based logical reasoning approach to predict movie ratings for the publicly available MovieLens dataset. We compare our results with other state-of-the-art ranking based approaches [11, 39, 24, 72].

7.2 Background

Typically, for the movie task the ratings are in a ordinal scale, say from 1 to 5 stars. Many machine learning approaches have been proposed which try to predict the *actual rating*. These include linear regression [90], nearest neighbor [24], and vector similarity based approaches [7]. However experimental results in [24] show that a ranking approach outperforms the above methods, on a set of metrics which evaluate the quality of the ordered list returned by the system. In a typical ranking formulation, for a given user, two movies are compared to determine which one is *preferred*. In general, a list of rated movies can always be decomposed down to a set of pairwise preference relations.

Consider a set of movies $\mathcal{X}$. For any user u and $(x, y) \in \mathcal{X} \times \mathcal{X}$ we interpret the *preference relation* $x \succeq_u y$ as '*user* u *prefers movie* x *over movie* y'. One way of describing preference relations is by means of a ranking function. A function $f_u : \mathcal{X} \to \mathbb{R}$ is a *ranking/scoring function* representing the preference relation $\succeq_u$ if $\forall x, y \in \mathcal{X}$, $x \succeq_u y \Leftrightarrow f_u(x) \geq f_u(y)$. The ranking function f_u provides a numerical score to the movies based on which the movies can be ordered. The ranking function is similar to the *utility function* used in microeconomic theory [62, 39], where utility is a measure of the satisfaction gained by consuming commodities. Various approaches have been proposed to learn ranking functions [11, 39, 24, 72]. Most of these use the ratings provided by other users as a feature vector for each movie. As mentioned earlier, there exist a large number of other cues that can help make a prediction for a given user's preference. However, a drawback of the above mentioned machine learning approaches is that it is not easy to exploit and incorporate these cues in an existing framework.

In the proposed bilattice based logical reasoning approach, we directly infer a proposition of the form $\mathit{prefer}(u, x, y)$. This proposition is the logical equivalent of $x \succeq_u y$. Bilattices are mathematical structures proposed by Ginsberg [35]. The use of bilattices for preference modeling has been theoretically explored by Arieli et al. [69]. The work described in this chapter is inspired by their use of bilattices for preference modeling but differs from it in the use of logical rules for inference and the fact that we have applied and evaluated this approach to a real world problem of movie ratings and compared with state-of-the-art approaches. In the past, we have also employed the bilattice based logical reasoning framework for a number of applications in the field of computer vision, especially to that of human activity recognition and identity maintenance [81] for automated visual surveillance.

Assume the following:

Rules:

$\phi[prefer(U, M1, M2) \leftarrow prefer(V, M1, M2), U \neq V, similar(U, V)] = \langle 0.8, 0.2\rangle$

$\phi[prefer(U, M1, M2) \leftarrow movie\_genre(M1, G1), movie\_genre(M2, G2), user\_prefer\_genre(U, G1, G2)] = \langle 0.6, 0.4\rangle$

$\phi[\neg prefer(U, M1, M2) \leftarrow prefer(V, M1, M2), U \neq V, dissimilar(U, V)] = \langle 0.4, 0.6\rangle$

Facts:

$\phi[prefer(2, 10, 20)] = \langle 0.625, 0.375\rangle$	$\phi[movie\_genre(10, 5)] = \langle 1, 0\rangle$	$\phi[prefer(3, 10, 20)] = \langle 1, 0\rangle$
$\phi[1 \neq 2] = \langle 1, 0\rangle$	$\phi[movie\_genre(20, 6)] = \langle 1, 0\rangle$	$\phi[1 \neq 3] = \langle 1, 0\rangle$
$\phi[similar(1, 2)] = \langle 0.64, 0.36\rangle$	$\phi[user\_prefer\_genre(5, 6)] = \langle 0.7, 0.3\rangle$	$\phi[dissimilar(1, 3)] = \langle 0.75, 0.25\rangle$

Inference is performed as follows:

$$\begin{aligned} cl(\phi)(prefer(1, 10, 20)) \quad = \quad & \langle 0, 0\rangle \vee \Big[\langle 0.8, 0.2\rangle \wedge \langle 0.625, 0.375\rangle \wedge \langle 1, 0\rangle \wedge \langle 0.64, 0.36\rangle\Big] \\ \oplus \quad & \langle 0, 0\rangle \vee \Big[\langle 0.6, 0.4\rangle \wedge \langle 1, 0\rangle \wedge \langle 1, 0\rangle \wedge \langle 0.7, 0.3\rangle\Big] \\ \oplus \quad & \neg\Bigg(\langle 0, 0\rangle \vee \Big[\langle 0.4, 0.6\rangle \wedge \langle 1, 0\rangle \wedge \langle 1, 0\rangle \wedge \langle 0.75, 0.25\rangle\Big]\Bigg) \\ = & \langle 0.32, 0\rangle \oplus \langle 0.42, 0\rangle \oplus \neg\langle 0.3, 0\rangle = \langle 0.6056, 0\rangle \oplus \langle 0, 0.3\rangle = \langle 0.6056, 0.3\rangle \end{aligned}$$

Figure 7.1: Example showing inference using closure within a $([0, 1]^2, \leq_t, \leq_k)$ bilattice

7.3 Recommender system

Rules can be defined within this bilattice framework to model cues that help predict user preference. Although the notion of preference is strictly binary, we consider the difference in the numerical value of the movie ratings between the two movies as an indicator of uncertainty of preference. We assume that greater the difference in rating between the two movies, lesser the uncertainty. We normalize this value to lie in the [0,1] unit interval. For example, if Cathy rated movie $M1$ as 1 and movie $M2$ as 5, it indicates that she definitely prefers $M2$ over $M1$ and a fact $\phi[prefer(cathy, M2, M1)] = \langle 1, 0 \rangle$ is asserted in the knowledge base. Similarly, if she rates movies $M3$ as 2 and $M4$ as 3, then we assume she only slightly prefers $M4$ over $M3$ and assert $\phi[prefer(cathy, M4, M3)] = \langle 0.625, 0.375 \rangle$[1]. Rules are specified similarly as $\phi(prefer(U, X, Y) \leftarrow \cdots) = \langle \gamma, \delta \rangle$. γ and δ are learnt as outlined in section 7.3.2. At evaluation time, given a query user, u, and a pair of movies (x, y) the system submits the query $prefer(u, x, y)$ to the logic program where support for and against it is gathered and finally combined into a single answer within the bilattice framework. Projecting the uncertainty value onto the $\langle 0, 1 \rangle - \langle 1, 0 \rangle$ axis, gives us the final degree of belief in the hypothesis.

7.3.1 Rules Specification

In our system, we have three kinds of rules.

[1] Note: if Cathy had rated both $M3$ and $M4$ as 2, then we would assume she does not prefer one over the other and assert $\phi[prefer(cathy, M4, M3)] = \langle 0.5, 0.5 \rangle$

User Similarity based rules: These rules capture a given user's preference based on his/her similarity/dissimilarity to other users.

Data correlation based rules: These rules capture correlations between different data variables, such as those between user age, gender, occupation, education, income, etc and properties of the movie such as genre, and year of release. Higher order correlations such as those between more than two variables like age, gender, and genre can also be encoded in such rules.

Personal preference based rules: These rules capture personal preferences of a given user for genre, year of release, actor etc.

All the rules described above have both positive and negative variants. It is important to note that rules gleaned from domain experts can be easily incorporated into this system and their weights learned as outlined in section 7.3.2.

7.3.2 Learning

Given a rule of the form $A \leftarrow B_1, B_2, \cdots, B_n$, a confidence value of

$$\big\langle \mathcal{P}(A|B_1, B_2, \cdots, B_n), \mathcal{P}(\neg A|B_1, B_2, \cdots, B_n) \big\rangle$$

is computed, where $\mathcal{P}(A|B_1, B_2, \cdots, B_n)$ is the fraction of times A is true when $B_1, B_2, \cdots, B_n$ is true. When using the probabilistic sum as a t-conorm, we need to uniformly scale down the rule weights to prevent uncertainties for all inferences from trivially hitting $\langle 1, 1 \rangle$. This scaling factor can be set to be proportional to the size of the feature vector used.

7.3.3 Generating proofs

The system provides a straightforward technique to generate proofs from its inference tree. Since all of the bilattice based reasoning is encoded as meta logical rules in a logic programming language, it is easy to add predicates that succeed when the rule fires and propagate character strings through the inference tree up to the root where they are aggregated and displayed. Such proofs can either be dumps of the logic program itself or be English text. In our implementation, we output the logic program as the proof tree. Only the relevant part of this proof tree can be displayed by thresholding on the computed uncertainty values.

7.4 Experiments

We use the MovieLens dataset [2] which contains approximately 1 million anonymous ratings for 3592 movies by 6040 users. The datset also has some demographic information on the users (gender, age, occupation, zip-code) and the genre and year of release of the movies. Ratings are made on a discrete scale of 1 to 5. Each user has rated at least 20 movies. The task is to predict the movie ratings for a user based on the ratings provided by other users. We removed any movies which have been rated by less than 20 people and any users who have rated less than 20 movies.

[2] Downloaded from `http://www.grouplens.org/`.

7.4.1 Algorithms compared

Experimental results in [24] show that a ranking approach outperforms the regression and classification based approaches. Hence we compare the performance of the proposed bilattice approach against the following ranking based approaches.

1. *RankNCG* [72] A simple linear ranking function is learnt which maximizes the number of pairwise agreements. The learning algorithm is based on a non-linear conjugate-gradient algorithm. The tolerance for the conjugate gradient procedure was set to 10^{-3}.

2. *RankNet* [11] A neural network which is trained using pairwise samples based on cross-entropy cost function.Training was done for around 500-1000 epochs. We used two versions of the RankNet: (a) *RankNet two layer* A two layer neural network with 10 hidden units; (b) *RankNet linear* A single layer neural network.

3. *RankSVM* [51, 39] A ranking function is learnt by training an SVM classifier [3] over pairs of examples. We used a linear kernel.

4. *RankBoost* [24] A boosting algorithm which combines a set of *weak* rankings. We used weak binary rankings as the ordering information provided by the features, boosted for 50-100 cycles.

For the proposed bilattice framework we used the weights learnt by the RankNCG procedure as the similarity/dissimilarity measure between users. The correlation

[3] SVM-light `http://svmlight.joachims.org/`

based rules employ correlations only between two variables.

7.4.2 Evaluation procedure

For each user we used 70% of the movies rated by him for training and the remaining 30% for testing. The feature vector for each movie consisted of the rating provided by first d other users. The users were ordered by the number of movies they have rated. Input preference relations for users who did not rate one or both movies we considered to be unknown or $\langle 0, 0 \rangle$. We did not impute missing movie ratings unlike in most other approaches compared. The results shown are averaged over 100 users each of who has rated on average 70.93 [± 12.14] movies.

7.4.3 Performance measure

In order to asses the quality of the rankings we use a generalized version of the *Wilcoxon-Mann-Whitney* (WMW) statistic [60, 27] defined as follows

$$\text{WMW}(f) = \frac{\sum_i \sum_j \mathbf{1}_{f(x_i) \geq f(x_j)}}{\sum_i \sum_j 1}, \tag{7.1}$$

where $\mathbf{1}_{a \geq b} = 1$ if $a \geq b$ and 0 otherwise and f is the ranking function learnt. The numerator counts the number of correct pairwise orderings. The denominator is the total number of pairwise preference relations available. The WMW is an estimate of $\Pr[prefer(u, x_i, x_j) | x_i \succeq_u x_j]$ for a randomly drawn pair of samples (x_i, x_j). This is a generalization of the area under the ROC curve. For a perfectly ranking function WMW=1, and for a completely random assignment WMW=0.5.

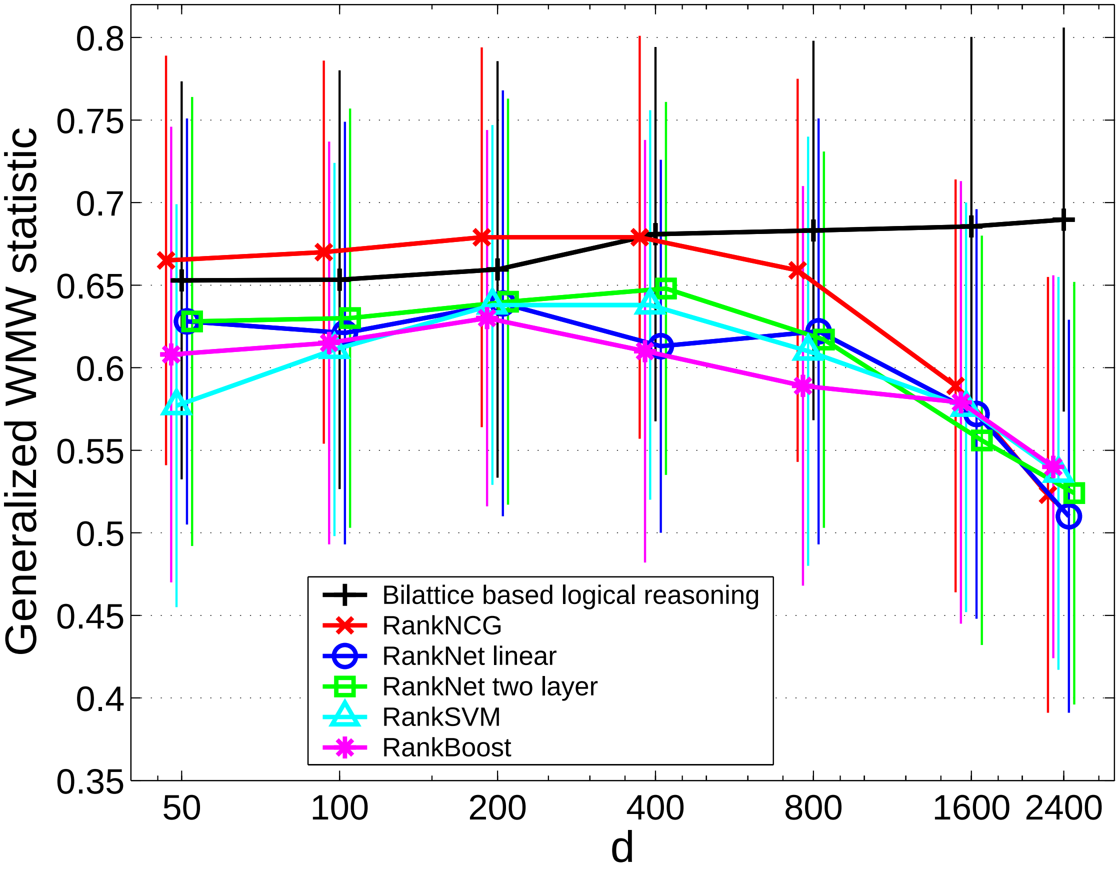


Figure 7.2: Plot of the mean WMW statistic along with the error bars(± one standard deviation) against the number of features (d) used.

7.4.4 Results

Figure 7.2 shows the mean WMW statistic along with the standard deviations for the different approaches. The following observations can be made.

(1) For small $d \leq 400$ the RankNCG approach shows the best performance followed by the proposed bilattice based framework. The remaining methods show very similar performance. The general trend is that as d increases the WMW statistic increases.

(2) However for large $d > 400$ it can be seen that as the number of features used is increased, the accuracy of the proposed approach increases, *while that of all other competing ranking based approaches drop.* One reason for this is that we have shown the results for the users who have rated only a few movies. In such cases the number of training examples is far less than the number of features used. It is known that maximum likelihood estimation (used in RankNCG) often fails when the number of features d is quite large. Also the optimization procedures used in the other ranking based approaches result in a lack of convergence when d is very large, possible due to numerical ill-conditioning.

While the ranking based approaches only take into account the similarity between a given user and the set of d users, the bilattice based logical reasoning approach exploits a much richer set of constraints and correlations and therefore gets better with increasing d.

(3) The standard deviation estimates for all the methods are roughly the same.

7.5 Conclusions

We have presented a bilattice based logical reasoning approach for modeling and predicting user preferences for the task of movie recommendations. Experiments indicate that this approach gives good results especially when the size of the feature vector is large compared to the number of training examples – a situation where the accuracy of other ranking based approaches drops. This is due to the fact that this approach can leverage a richer set of constraints, correlations and historical user preferences.

Chapter 8

Discussions

Reasoning under uncertainty is central to many real world applications and naturally it has been well studied. AI research tackling this problem falls broadly into two categories viz. *extensional* and *intensional*. Extensional approaches, also known as production systems, rule-based or procedure based systems, treat uncertainty as a generalized truth value attached to formulas and computes the uncertainty of any formula as a function of the uncertainties of its sub-formulas. In intensional approaches, also known as declarative or model based, uncertainty is attached to the "states of affairs" or subsets of "possible worlds". Extensional systems tend to be computationally efficient but semantically inadequate while intensional systems are semantically clear but computationally inefficient. The trade-off between semantic clarity and computational efficiency has been the subject of intense research in the past.

8.1 Extensional approaches

As mentioned earlier, extensional systems work by treating uncertainty as a generalized truth value attached to the formulas and computing uncertainty of any formula as a function of the truth values of its sub-formulas. E.g. uncertainty of the conjunction $A \wedge B$ is given by some function (min or times) of the uncertainty of

A and B individually. Rules in such systems are interpreted as licenses for certain symbolic activities. E.g $A \xrightarrow{m} B$ means “if you see A then you are given the license to update certainty of B by an amount that is a function of the rule strength m”. Rules are interpreted as a summary of the past performance of the agent and represent summaries of past information

8.1.1 Computational Merits

These systems derive their computational merits from the principle of modularity which is explained next. In extensional systems, given the credibility of each rule and the certainty of the premises, the same combination function applies uniformly to two rules in the system, regardless of what the other rules might be in the knowledge base. This uniformity mirrors modularity of inference rules in such systems. E.g. $A \rightarrow B$ has the following interpretation: If you see A anywhere in the KB, then *regardless of what other things* the KB contains, and *regardless of how A was derived*, you are given the license to assert B and add it to the KB. This combination of *Locality* (regardless of other things) and *Detachment* (regardless of how it was derived) constitutes the principle of *Modularity.* In such systems, uncertainty is updated as follows: Given rule $A \xrightarrow{m} B$, if certainty of A changes by ΔA, then the current certainty of B changes by ΔB which is a function of rule credibility m, ΔA and current certainty of B.

8.1.2 Semantic Issues

The computational merits mentioned in the previous section come at a price. The problems arise in several ways, mainly:

1. Bidirectional Inference

2. Limitations of modularity

3. Correlated evidence

Bidirectional Inferences Plausible reasoning requires that both predictive as well as diagnostic components of reasoning be used. if $A \rightarrow B$, then finding B to be true makes A more credible (abductive reasoning). This requires reasoning both ways. Extensional systems do not allow such bi-directional inference i.e. reasoning from both A to B and B to A. To implement this in extensional systems, one has to explicitly specify the reverse rule, possibly risking creation of a cycle that can cause evidence to be cyclically amplified until both cause and effect are completely certain with no apparent factual justification.

Removing the predictive component prevents system from exhibiting another important pattern of plausible reasoning called explaining away: if $A \rightarrow B$ and $C \rightarrow B$ and B is true, then finding C is true makes A less credible. To exhibit this kind of reasoning, the system must use bidirected inferences; from evidence to hypothesis and from hypothesis to evidence. While it might be possible to get around this problem by exhaustively listing all possible exceptions, to restore explaining away (without the danger of circular reasoning), any system that does that sacrifices on principles of modularity. Inversely, any system that updates beliefs modularly and treats all rules equally is bound to defy patterns of plausible reasoning

Limits of Modularity In extensional systems, detachment can create problems. In deductive logic the following holds true: $A \rightarrow B$ and $B \rightarrow C \Rightarrow A \rightarrow C$. In other words, finding evidence for A leads us to conclude C by chaining. Derived evidence B triggers the rule $B \rightarrow C$ with the same rigor as would a directly observed proposition. However consider the case, "*ground_is_wet* → *it_rained*" and "*sprinkler_is_on* → *ground_is_wet*". In this case, if an extensional system is told that *sprinkler_is_on*, it will conclude that *it_rained*. This is incorrect and infact finding that the sprinkler was on should only reduce the likelihood that it rained.

Correlated Evidence Due to locality, extensional systems do not store information on how a proposition was derived. As a result, they risk treating correlated evidence as independent. E.g. consider a situation where someone hears a piece of news independently from the radio, television as well as the newspapers. Since from his point of view, the sources are independent, his belief in the veracity of the piece of news should be very high. However, if that person were to realize later that all the three sources got their information from the same source, then his belief in the piece of news should decrease. This can never happen in extensional systems as they treat each source of information completely independently of the others.

8.2 Intensional approaches

As mentioned earlier, in intensional systems, uncertainty is attached to "states of affairs" or subsets of "possible worlds". Such systems compute the uncertainty of any formula by combining sets of worlds by set theory operations. E.g. the probability $P(A \wedge B)$ is given by the weight assigned to the intersection of two sets of worlds, one in which A is true and the other in which B is true, but $P(A \wedge B)$ cannot be determined by P(A) and P(B) alone. In such systems, rules represent elastic constraints about the world. E.g. in Dempster-Shafer formalism, $A \xrightarrow{m} B$ asserts the set of worlds in which A and B hold simultaneously has low likelihood and hence should be excluded with probability m. In Bayesian formalism, $A \xrightarrow{m} B$ is interpreted as conditional probability $P(B|A) = m$. Rules here represent summaries of factual or empirical information.

8.2.1 Semantic Merits

The primary benefit provided by the intensional approach is the ability to perform plausible reasoning. Unlike in extensional systems, interpreting rules as conditional probabilities $P(B|A)$ does not give a license to do anything. The meaning of $P(B|A) = m$ is that if you know A and A is the ONLY thing you know then you can attach to B a probability of m. As soon as other facts K appear, the license to assert $P(B) = m$ is automatically revoked and we need to look up $P(B|A, K)$. Moreover, given that B is true, increases the credibility of A being true. All of the issues mentioned in 8.1.2, including bidirectional inference and capacity to handle correlated evidence are addressed in these approaches.

8.2.2 Computational Problems

The downside of accepting a strict approach to computing uncertainties is computational intractability. Attempting to compute $P(B|A_1, A_2, \cdots, A_n)$ for all propositions A_i is a computationally intractable task. Due to this requirement, probability statements leave such systems impotent, unable to initiate any computation, unless it is explicitly verified that everything else in the knowledge base is irrelevant. Verification of irrelevancy is therefore crucial to intensional systems and also the cause for its computational problems.

8.3 Comparisons

In this section we will note some similarities and differences between the bilattice based logical reasoning approach employed in this thesis and the Bayesian networks approach. The former is an example of an extensional approach while the latter that of an intensional one.

8.3.1 Theoretical comparison

Comparisons between the two systems can be made at two levels. The first one is at the level of the inference structure and the second is in the inference process itself. The inference structure for the logical approach is specified by the logical formulae. Given the rules and the facts, the rules are applied to the facts in an attempt to unify the unbounded variables in the rules with ground atoms (facts). This process of unification produces a proof tree. This proof tree corresponds closely to the graphical model used in Bayesian Networks. The second level at which correspondence needs to be established is at the level of the inference procedure. In Bayesian networks, once the graphical model is acquired, conditional probability tables (CPTs) need to be learnt at each node. The CPT captures the statistics of the given node conditioned on its parents. Specifically, for binary variables, the CPT captures, for each variable, the probabilities for each of its states, conditioned on all possible states its parents can be in. Clearly, if a given node has k parents, then the CPT contains 2^{k+1} elements.

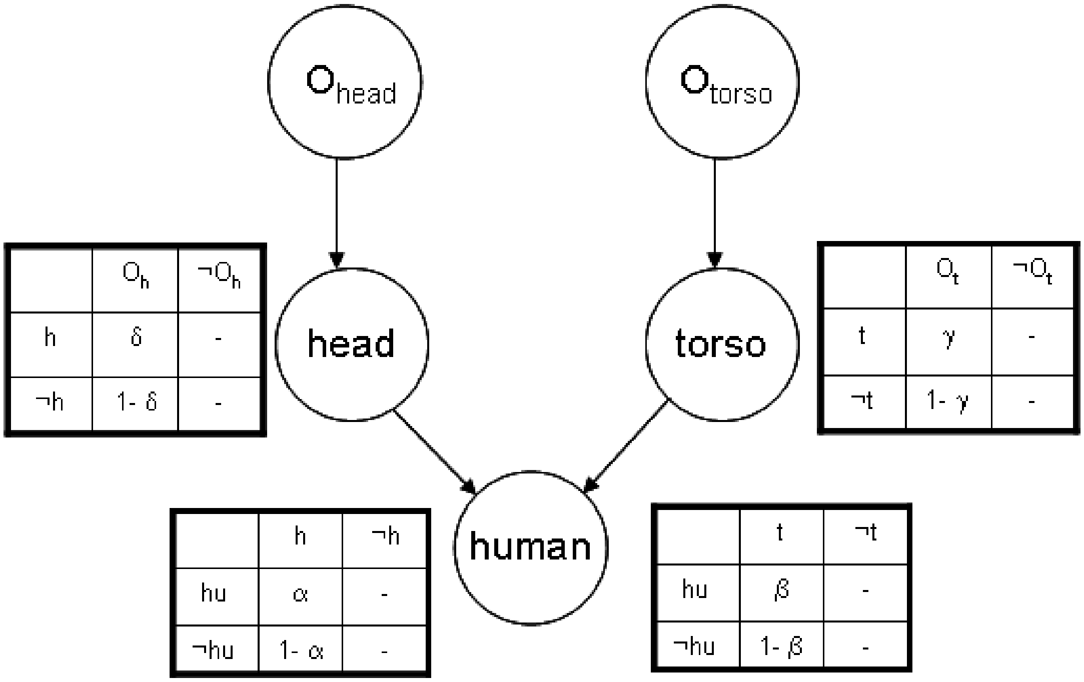
Inference in a bayesian network occurs by integrating the joint probability function over the variables to be eliminated. This integration requires that every element of the CPT, describing the variable to be eliminated, be considered. The complexity of such an operation is $O(n2^k)$ assuming an average of k parents for each of n nodes. On the other hand, in logical system such as the one employed in this thesis, inference is defined in terms of the closure over the truth assignment. This process takes the uncertainty value assigned to each node of the graphical model and combines them using the join and meet operator (however they may be defined) along both the information and truth axis of the bilattice. This operation visits each node only once and therefore the complexity of this operation is $O(n)$, where n is the number of nodes in the graph.

The computation gain in using logical approaches takes the runtime from $O(n2^k)$ to $O(n)$. However, this computational gain comes at a cost. The cost is that using the bilattice based logical reasoning framework, it is not possible to perform arbitrary queries. The important question that remains unanswered is, if we restrict ourselves to only making forward queries, i.e. given some observations, we are only interested in their consequences, then does jettisoning the extra machinery that helps us make arbitrary queries, actually hurt us? Note that in all of the examples listed in this thesis, we have always been performing forward inferences. Given the observations, we have inferred the activity. Given the observations, we have inferred whether there exists human in the image or not. In such applications, it seems we are not interested in e.g. inferring the likelihood of an observation given that there exists a human in the image. In the literature, there exist approximations that exploit this constraint to reduce the complexity of Bayesian inference from $O(n2^k)$ to $O(n)$. Notable among such approaches are Naive Bayes and Noisy-OR. We will computationally compare the bilattice based logical reasoning approach with the Noisy-OR approach since it most closely resembles it.

8.3.2 Algebraic comparison

Consider the problem of detecting humans in static images from parts. Assume that we have designed two low level parts based detectors to detect the presence of heads and torsos. Based on these two detectors, we can write two rules as follows:

$$\begin{aligned} human(X,Y,S) &\leftarrow head(X,Y,S). \\ human(X,Y,S) &\leftarrow torso(X,Y,S). \end{aligned} \tag{8.1}$$



	O_h	$\neg O_h$
h	δ	-
¬h	1- δ	-

	O_t	$\neg O_t$
t	γ	-
¬t	1- γ	-

	h	¬h
hu	α	-
¬hu	1- α	-

	t	¬t
hu	β	-
¬hu	1- β	-

Figure 8.1: Simple graphical model for human detection from parts.

These two rules capture the information that if we “see” a head at a particular location, then we can infer that there exists a human at that location. Similarly if we “see” a torso at a particular location, then we can make a similar inference. The graphical model for these set of rules within a statistical framework is depicted in figure 8.1. In the Noisy-OR approach, given three random variable A, B and C,

$$P(A|B,C) = 1 - (1 - P(A|B))(1 - P(A|C))$$

This is the key simplifying step that reduces complexity from $O(n2^k)$ to $O(n)$. Given the model in figure 8.1, we can then ask the question, what is the probability of seeing a human given the observations.

$$
\begin{aligned}
P(human|O_h, O_t) \quad = \quad & \Sigma_{head}\Sigma_{torso}P(human|head, torso)P(head|O_h)P(torso|O_t) \\
& = P(human|head, torso)P(head|O_h)P(torso|O_t) \\
& +P(human|\neg head, torso)P(\neg head|O_h)P(torso|O_t) \\
& +P(human|head, \neg torso)P(head|O_h)P(\neg torso|O_t) \\
& +P(human|\neg head, \neg torso)P(\neg head|O_h)P(\neg torso|O_t) \\
& = [1-(1-\alpha)(1-\beta)]\gamma\delta + \beta(1-\delta)\gamma + \alpha\delta(1-\gamma) + 0 \\
& = \alpha\delta + \beta\gamma - \alpha\beta\gamma\delta \qquad (8.2)
\end{aligned}
$$

In the bilattice based logical reasoning approach, if we assume a continuous bilattice as described in chapter 6, and also that the t-norm is set to the product and t-conorm to the probabilistic sum, then we get:

$$
\begin{aligned}
\phi[human(X,Y,S) \leftarrow head(X,Y,S)] &= \langle \alpha, 1-\alpha \rangle. \\
\phi[human(X,Y,S) \leftarrow torso(X,Y,S)] &= \langle \beta, 1-\beta \rangle. \qquad (8.3)
\end{aligned}
$$

$$\begin{aligned}
\phi(human) &= \phi(\bot) \vee [\phi(head) \wedge \phi(human \leftarrow head)] \\
& \oplus \phi(\bot) \vee [\phi(torso) \wedge \phi(human \leftarrow torso)] \\
& = \langle 0,0\rangle \vee [\langle \delta, 1-\delta\rangle \wedge \langle \alpha, 1-\alpha\rangle] \\
& \oplus \langle 0,0\rangle \vee [\langle \gamma, 1-\gamma\rangle \wedge \langle \beta, 1-\beta\rangle] \\
& = \langle 0,0\rangle \vee [\langle \alpha\delta, \alpha+\delta-\alpha\delta\rangle \oplus \langle 0,0\rangle \vee [\langle \beta\gamma, \beta+\gamma-\beta\gamma\rangle \\
& = \langle \alpha\delta, 0\rangle \oplus \langle \beta\gamma, 0\rangle \\
& = \langle \alpha\delta + \beta\gamma - \alpha\beta\gamma\delta, 0\rangle \qquad (8.4)
\end{aligned}$$

It can be seen that the expressions in equations 8.2 and 8.4 are exactly equal. This indicates that a bilattice based logical reasoning approach will always agree with a Noisy-OR formulation of the same problem, as long as the rules encode positive information.

This is however not the case when there exists a rule which infers an explicit negation. Consider the rules

$$\begin{aligned}
\phi[human(X,Y,S) \leftarrow head(X,Y,S)] = \langle \alpha, 1-\alpha\rangle. \\
\phi[human(X,Y,S) \leftarrow torso(X,Y,S)] = \langle \beta, 1-\beta\rangle. \\
\phi[human(X,Y,S) \leftarrow not(legs(X,Y,S))] = \langle \eta, 1-\eta\rangle. \qquad (8.5)
\end{aligned}$$

The third rule in the equation above captures the information that if we cannot locate the legs of a human, then the hypothesis is probably not a human. Incorporating this information in the bilattice based logical reasoning framework is straightforward and we compute the uncertainty value for human to be

$$\phi(human) = \langle \alpha\delta + \beta\gamma - \alpha\beta\gamma\delta, \omega\eta \rangle \tag{8.6}$$

assuming $\phi[not(legs(X, Y, S))] = \langle \omega, 1 - \omega \rangle$. It is not clear how the third rule can be trivially incorporated within the Noisy-OR framework while maintaining the computational complexity of $O(n)$ for inference.

Chapter 9

Conclusions

The primary objective of an automated visual surveillance system is to observe and understand human behavior and report unusual or potentially dangerous activities/events in a timely manner. Successfully accomplishing this task requires the system to (a) take visual input from possibly multiple cameras, (b) identify objects of interest (c) classify these objects into known types (d) track the objects while they are within the field of regard of the cameras, (e) log the occurrence of basic events such as object interactions, and finally (f) employ these basic events to reason about occurrence of various activities of interest, possibly spanning large intervals of time. This task, however, is made challenging by the ubiquitous presence of uncertainty within all components of this pipeline.

In this thesis, we have proposed a high level logical reasoning approach draws heavily upon human like reasoning and reasons explicitly about visual as well as non-visual information to solve classification problems. This framework can combine, in a principled manner, high level contextual information with low level image processing primitives to interpret visual information and make decision under the uncertainties inherent in visual systems. We applied this framework to the problems of human detection, identity maintenance, occlusion handling and activity recognition within the domain of computer vision. To demonstrate the generality of this approach, also applied it to a problem in the field of machine learning, that of collaborative filtering. In all these cases, the proposed approach gave results that compared favorably to the state-of-the-art approaches while giving us the benefits of, among others, reduced complexity of inference, greater power of expression on account of logic programming, and the power to generate justifications for every decision.

Bibliography

[1] CAVIAR homepage:http://homepages.inf.ed.ac.uk/rbf/caviar/.

[2] Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *J. Symb. Log.*, 50(2):510–530, 1985.

[3] Ofer Arieli, Chris Cornelis, Glad Deschrijver, and Etienne Kerre. Bilattice-based squares and triangles. *Lecture Notes in Computer Science: Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 563–575, 2005.

[4] C BenAbdelkader, R. Cutler, and L. Davis. Motion-based recognition of people in eigengait space. In *Proc of Intl. Conf. on Auto Face and Gesture Recogtn*, page 267, 2002.

[5] M. Brand and V. Kettnaker. Discovery and segmentation of activities in video. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):844–851, Aug 2000.

[6] M Brand, N Oliver, and A Pentland. Coupled hidden markov models for complex action recognition. In *Proc. CVPR*, pages 994–999, 1997.

[7] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

[8] F. Bremond and M. Thonnat. A context representation for surveillance systems. In *ECCV Worshop on Conceptual Descriptions from Images*, April 1996.

[9] Gerhard Brewka. Adding priorities and specificity to default logic. In *JELIA '94: Proceedings of the European Workshop on Logics in Artificial Intelligence*, pages 247–260. Springer-Verlag, 1994.

[10] G. J. Brostow and I. A. Essa. Motion based decompositing of video. *International Conference on Computer Vision*, 2001.

[11] C.J.C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceeding of the 22nd International conference on Machine Learning*, 2005.

[12] H. Buxton and S. Gong. Advanced Visual Surveillance using Bayesian Networks. In *International Conference on Computer Vision*, Cambridge, Massachusetts, June 1995.

[13] B. Clarkson and A. Pentland. Unsupervised clustering of ambulatory audio and video. Technical Report 471, MIT Media Lab, Perceptual Computing Group., 1998.

[14] A. G. Cohn, D. Magee, A. Galata, D. Hogg, and S. M. Hazarika. Towards an architecture for cognitive vision using qualitative spatio-temporal representations and abduction. In *Spatial Cognition III*, 2002.

[15] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. *Computer Vision and Pattern Recognition, Proceedings. IEEE Conference on*, 2:142–149, 2000.

[16] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR05*, pages I: 886–893, 2005.

[17] O. Dekel, C. Manning, and Y. Singer. Log-linear models for label ranking. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[18] Didier Dubois, Jérôme Lang, and Henri Prade. Automated reasoning using possibilistic logic: Semantics, belief revision, and variable certainty weights. *IEEE Trans. Knowl. Data Eng.*, 6(1):64–71, 1994.

[19] Marcelo A. Falappa, Gabriele Kern-Isberner, and Guillermo Ricardo Simari. Explanations, belief revision and defeasible reasoning. *Artif. Intell.*, 141(1/2):1–28, 2002.

[20] P.F. Felzenszwalb. Learning models for object recognition. In *CVPR01*, pages I:1056–1062, 2001.

[21] J. Fernyhough, A. Cohn, and D. Hogg. Building qualitative event models automatically from visual input. *Proc. ICCV*, pages 350–355, 1998.

[22] J. Forbes, T. Huang, K. Kanazawa, and S. J. Russell. The BATmobile: Towards a bayesian automated taxi. In *IJCAI*, pages 1878–1885, 1995.

[23] Y. Freund, R. Iyer, and R. Schapire. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

[24] Y. Freund, R. Iyer, and R. Schapire. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

[25] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journ. of Comp. and System Sciences*, 55:119–139, 1997.

[26] André Fuhrmann and Michael Morreau, editors. *The Logic of Theory Change, Workshop, Konstanz, FRG, October 13-15, 1989, Proceedings*, volume 465 of *Lecture Notes in Computer Science*. Springer, 1991.

[27] G. Fung, R. Rosales, and B. Krishnapuram. Learning rankings via convex hull separation. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*. MIT Press, Cambridge, MA, 2006.

[28] A. Galata, N. Johnson, and D. Hogg. Learning variable-length markov models of behavior. *Comput. Vis. Image Underst.*, 81(3):398–413, 2001.

[29] Peter Gärdenfors. Belief revisions and the ramsey test for conditionals. *The Philosophical Review*, 95:81–93, 1986.

[30] D. Gavrila. Pedestrian detection from a moving vehicle. In *ECCV00*, pages II: 37–49, 2000.

[31] D. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *ICCV99*, pages 87–93, 1999.

[32] R. Gerber, H. Nagel, and H. Schreiber. Deriving textual descriptions of road traffic queues from video sequences. In *Proceedings of the 15th ECAI'2002 Lyon France*. IOS Press, July 2002.

[33] M. Ghallab. On chronicles: Representation, on-line recognition and learning. *Principles of Knowledge Representation and Reasoning*, pages 597–606, November 1996.

[34] M. L. Ginsberg. Multi-valued logics: a uniform approach to reasoning in artificial intelligence. *Computational Intelligence*, 4:265–316, 1988.

[35] Matthew L. Ginsberg. Multivalued logics: A uniform approach to inference in artificial intelligence. *Computational Intelligence*, 4(3):256–316, 1992.

[36] Peter Grdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. MIT Press, 1988.

[37] Sven Ove Hansson. Ten philosophical problems in belief revision. *J. Log. Comput.*, 13(1):37–49, 2003.

[38] I. Haritaoglu, D. Harwood, and L. Davis. W4: A real time system for detecting and tracking people. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 962, 1998.

[39] R. Herbrich, T. Graepel, P. Bollmann-Sdorra, and K. Obermayer. Learning preference relations for information retrieval. *ICML-98 Workshop: Text Categorization and Machine Learning*, pages 80–84, 1998.

[40] R. Herbrich, T. Graepel, and K. Obermayer. *Advances in Large Margin Classifiers*, chapter Large margin rank boundaries for ordinal regression, pages 115–132. MIT Press, 2000.

[41] Derek Hoiem, Alexei Efros, and Martial Hebert. Putting objects in perspective. In *CVPR*, 2006.

[42] S. Hongeng, F. Brémond, and R. Nevatia. Representation and optimal recognition of human activities. *Proc. of the IEEE CVPR*, June 2000.

[43] S. Hongeng, R. Nevatia, and F. Bremond. Video-based event recognition: activity representation and probabilistic recognition methods. *CVIU*, 96(2):129–162, November 2004.

[44] J. Horty. Skepticism and floating conclusions. *Artificial Intelligence*, 135:55–72, 2002.

[45] C. Huang, H. Al, B. Wu, and S. Lao. Boosting nested cascade detector for multi-view face detection. In *ICPR04*, pages II: 415–418, 2004.

[46] S. S. Intille and A. F. Bobick. A framework for recognizing multi-agent action from visual evidence. In *Proceedings of the sixteenth NCAIIAAI*, pages 518–525, 1999.

[47] Stephen S. Intille and Aaron F. Bobick. Closed-world tracking. *IEEE Conf. International Conference on Computer Vision*, pages 672–678, 1995.

[48] M. Isard and J. MacCormick. Bramble: A bayesian multiple-blob tracker. *International Confernce on Computer Vision*, pages 34–41, 2001.

[49] Y. A. Ivanov and A. F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):852–872, 2000.

[50] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:1296–1311, 2003.

[51] T. Joachims. Optimizing search engines using clickthrough data. *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, 2002.

[52] Franced L. Johnson and Stuart C Shapiro. Formalizing a deductiveli open belief space. Cse technical report 2000-02, Department of Computer Science, State University of New York at Buffalo, January 2000.

[53] N. Jojic and B.J. Frey. Learning flexible sprites in video layers. *IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[54] S. Khan and M. Shah. Tracking people in presence of occlusion. *Asian Conference on Computer Vision*, 2004.

[55] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis. Background modeling and subtraction by codebook construction. *IEEE ICIP*, 2004.

[56] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. *European Conference on Computer Vision*, pages 189–196, 1994.

[57] Robert Koons. Defeasible reasoning. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Spring 2005.

[58] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer. Multi-camera multi-person tracking for easyliving. *Proc. 3rd IEEE Intl Workshop on Visual Surveillance*, 2000.

[59] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *IEEE CVPR'05 in , San Diego, CA*, pages 878–885. sp, may 2005.

[60] H. B. Mann and D. R. Whitney. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50–60, 1947.

[61] J. S. Marques, P. M. Jorge, A. J. Abrantes, and J. M. Lemos. Tracking groups of pedestrians in video sequences. *IEEE Workshop on Multi-Object Tracking*, 2001.

[62] A. Mas-Colell, M.D. Whinston, and J.R. Green. *Microeconomic theory.* Oxford University Press, New York, 1995.

[63] J. McCarthy. Artificial intelligence, logic and formalizing common sense. *Philosophical Logic and Artificial Intelligence*, 1989.

[64] S. J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. *Computer Vision and Image Understanding*, 80:42–56, 2000.

[65] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *ECCV*, May 2004.

[66] A. Mohan, C.P. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *PAMI*, 23(4):349–361, April 2001.

[67] C. Nakajima, M. Pontil, B. Heisele, and T. Poggio. Full-body person recognition system. *Pattern Recognition*, 36(9):1997–2006, 2003.

[68] B. Nebel. Belief revision and default reasoning: Syntax-based approaches. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, pages 301–311. Morgan Kaufmann, 1991.

[69] O.Arieli, C.Cornelis, and G.Deschrijver. Preference modeling by rectangular bilattices. *Proc. 3rd International Conference on Modeling Decisions for Artificial Intelligence (MDAI'06)*, (3885):22–33, April 2006.

[70] C. Papageorgiou, T. Evgeniou, and T. Poggio. A trainable pedestrian detection system. *Intelligent Vehicles*, pages 241–246, October 1998.

[71] John L. Pollock. Defeasible reasoning. *Cognitive Science*, 11(4):481–518, 1987.

[72] V. C. Raykar, R. Duraiswami, and B. Krishnapuram. A fast algorithm for learning large scale preference relations. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*, 2007.

[73] J. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. *International Conference on Computer Vision*, 1995.

[74] R. Reiter. A logic for default reasoning. *Readings in nonmonotonic reasoning*, pages 68–93, 1987.

[75] N. A. Rota and M. Thonnat. Activity recognition from video sequences using declarative models. *14th ECAI 2000 Berlin Germany*, August 2000.

[76] Hans Rott. A nonmonotonic conditional logic for belief revision. In Fuhrmann A. and M. Morreau, editors, *The Logic of Theory Change, Workshop, Lecture Notes in Artificial Intelligence, Volume 465*, Konstanz, FRG, October 1989. Springer Verlag.

[77] B. Schweizer and A. Sklar. Associative functions and abstract semigroups. *Publ. Math. Debrecen*, 1963.

[78] A. Senior, A. Hampapur, Y.-L. Tian, L. Brown, S. Pankanti, and R. Bolle. Appearance models for occlusion handling. *In IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, 2001.

[79] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of Human Factors in Computing Systems CHI'95*, 1995.

[80] V. Shet, D. Harwood, and L. Davis. VidMAP: Video Monitoring of Activity with Prolog. *IEEE International Conference on Advanced Video and Signal based Surveillance*, pages 224–229, 2005.

[81] Vinay Shet, David Harwood, and Larry Davis. Multivalued Default Logic for Identity Maintenance in Visual Surveillance. *European Conference on Computer Vision*, IV:119–132, 2006.

[82] Gerardo I. Simari and Marcelo A. Falappa. Non prioritized belief revision with ansprolog*. *VI Workshop of Researchers in Computer Science*, 2004.

[83] J. Sochman and J. Matas. Waldboost – learning for time constrained sequential detection. CVPR 2005.

[84] T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. In *Proc of the Intl Symposium on Computer Vision*, 1995.

[85] H. Tao, H. S. Sawhney, and R. Kumar. Object tracking with bayesian estimation of dynamic layer representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):75–89, 2002.

[86] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. IEEE Conference on Computer Vision and Pattern Recognition (CVPR'01), 2001.

[87] P. Viola, M.J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *ICCV03*, pages 734–741, 2003.

[88] V. Vu, F. Bremond, and M. Thonnat. Automatic video interpretation: A novel algorithm for temporal scenario recognition. *The Eighteenth IJCAI '03*, August 2003.

[89] G. Wei, V. Petrushin, and A. Gershman. Multiple-camera people localization in a cluttered environment. *The 5th International Workshop on Multimedia Data Mining*, 2004.

[90] H. Will, M. Stead, L. amd Rosenstein, and G. Furnas. Recommending and evaluating choices in a virtual community of use. In *Proceedings of Human Factors in Computing Systems CHI'95*, 1995.

[91] A. D. Wilson and A. F. Bobick. Recognition and interpretation of parametric gesture. In *ICCV*, pages 329–336, 1998.

[92] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.

[93] Bo Wu and Ram Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. *ICCV*, Oct 2005. Beijing.

[94] Y. Wu, T. Yu, and G. Hua. Tracking appearances with occlusions. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2003.

[95] R. Yan and A. Hauptmann. Efficient margin-based rank learning algorithms for information retrieval. In *International Conference on Image and Video Retrieval (CIVR'06)*, 2006.

[96] K. Yoon, D. Harwood, and L. Davis. Appearance-based person recognition using color-path/length profile. Unpublished work.

[97] T. Zhao and R. Nevatia. Bayesian human segmentation in crowded situations. *CVPR*, 2:459–466, 2003.

[98] S Zhou, V Krueger, and R Chellappa. Probabilistic recognition of human faces from video. *Comput. Vis. Image Underst.*, 91(1-2):214–245, 2003.

[99] Y. Zhou and H. Tao. A background layer model for object tracking through occlusion. *International Conference on Computer Vision*, 2003.

[100] Q. Zhu, M.C. Yeh, K.T. Cheng, and S. Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *CVPR06*, pages II: 1491–1498, 2006.