ABSTRACT

Title of dissertation: ANALYZING STRUCTURED SCENARIOS BY TRACKING PEOPLE AND THEIR LIMBS

Vlad I. Morariu, Doctor of Philosophy, 2010

Dissertation directed by: Professor Larry S. Davis
Department of Computer Science

The analysis of human activities is a fundamental problem in computer vision. Though complex, interactions between people and their environment often exhibit a spatio-temporal structure that can be exploited during analysis. This structure can be leveraged to mitigate the effects of missing or noisy visual observations caused, for example, by sensor noise, inaccurate models, or occlusion. Trajectories of people and their hands and feet, often sufficient for recognition of human activities, lead to a natural qualitative spatio-temporal description of these interactions.

This work introduces the following contributions to the task of human activity understanding: 1) a framework that efficiently detects and tracks multiple interacting people and their limbs, 2) an event recognition approach that integrates both logical and probabilistic reasoning in analyzing the spatio-temporal structure of multi-agent scenarios, and 3) an effective computational model of the visibility constraints imposed on humans as they navigate through their environment. The tracking framework mixes probabilistic models with deterministic constraints and

uses AND/OR search and lazy evaluation to efficiently obtain the globally optimal solution in each frame. Our high-level reasoning framework efficiently and robustly interprets noisy visual observations to deduce the events comprising structured scenarios. This is accomplished by combining First-Order Logic, Allen's Interval Logic, and Markov Logic Networks with an event hypothesis generation process that reduces the size of the ground Markov network. When applied to outdoor one-on-one basketball videos, our framework tracks the players and, guided by the game rules, analyzes their interactions with each other and the ball, annotating the videos with the relevant basketball events that occurred. Finally, motivated by studies of spatial behavior, we use a set of features from visibility analysis to represent spatial context in the interpretation of human spatial activities. We demonstrate the effectiveness of our representation on trajectories generated by humans in a virtual environment.

ANALYZING STRUCTURED SCENARIOS BY TRACKING PEOPLE AND THEIR LIMBS

by

Vlad I. Morariu

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
2010

Advisory Committee:
Professor Larry S. Davis, Chair/Advisor
Professor Ramani Duraiswami
Professor Lise Getoor
Professor David W. Jacobs
Professor Min Wu

Acknowledgments

First, I would like to thank my advisor, Professor Larry Davis, who provided valuable guidance and feedback but also gave me the independence to freely explore my research ideas. I would also like to thank David Harwood, for the meaningful discussions that began my work on tracking people and their limbs, and Professor Ramani Duraiswami, for the thoughtful conversations related to my research.

I am grateful for the support that my wife and family have provided throughout the ups and downs of the Ph.D. process. My parents, Viorel and Mioara, and my sister, Elena, patiently listened to me recount my successes and failures—and, at times, overly detailed technical explanations. I am thankful for their love and friendship. My wife, Johanna, had the privilege (or burden?) of hearing about every step of the process. The encouragement that she provided, accompanied by delicious meals she prepared (and often shared with colleagues), helped me through the tough times and made the whole experience much more enjoyable.

My friends and colleagues, Ani, William, Ryan, Abhinav, Behjat, Radu, and Brandyn, have made my graduate experience memorable and fun through discussions (research and otherwise), tea and coffee breaks, foosball games, and evening gatherings. Shiv Naga Prasad provided valuable contributions to the visibility work presented in this dissertation. Other colleagues also enriched my experience: Arpit, Choi, Huimin, Hyungtae, Tom, Zhuolin, Balaji, Zhe, Vinay, Mohamed, and Son.

Finally, I thank my committee members for taking time out of their busy schedules to make the final step of my doctorate possible.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The analysis of human activities is a fundamental problem in computer vision. Though complex, interactions between people and their environment often exhibit a spatio-temporal structure that can be exploited during analysis. This structure can be leveraged to mitigate the effects of missing or noisy visual observations caused, for example, by sensor noise, inaccurate models, or occlusion. Trajectories of people and their hands and feet, often sufficient for recognition of human activities, lead to a natural qualitative spatio-temporal description of these interactions.

To provide a motivating example, figure 1.1 illustrates the analysis of one-on-one basketball, a scenario in which the rules of the game impose a strong spatio-temporal structure on the events that occur. In this example, the task is to annotate the basketball events that occurred (e.g., *check*, *dribble series*, *shot*) and their effects on properties of the game (e.g., *possession*). In order to produce the final result, a trajectory-based approach involves the following intermediate tasks. First, locations of people and objects are obtained from the video. Based on these trajectories, relationships between people and objects can be obtained. For example, in the basketball scenario, it is useful to know that a player (or the player's hand) is near the ball. Relationships between people and objects to their environment are also obtained from trajectories. In the motivating example, it may be useful to know

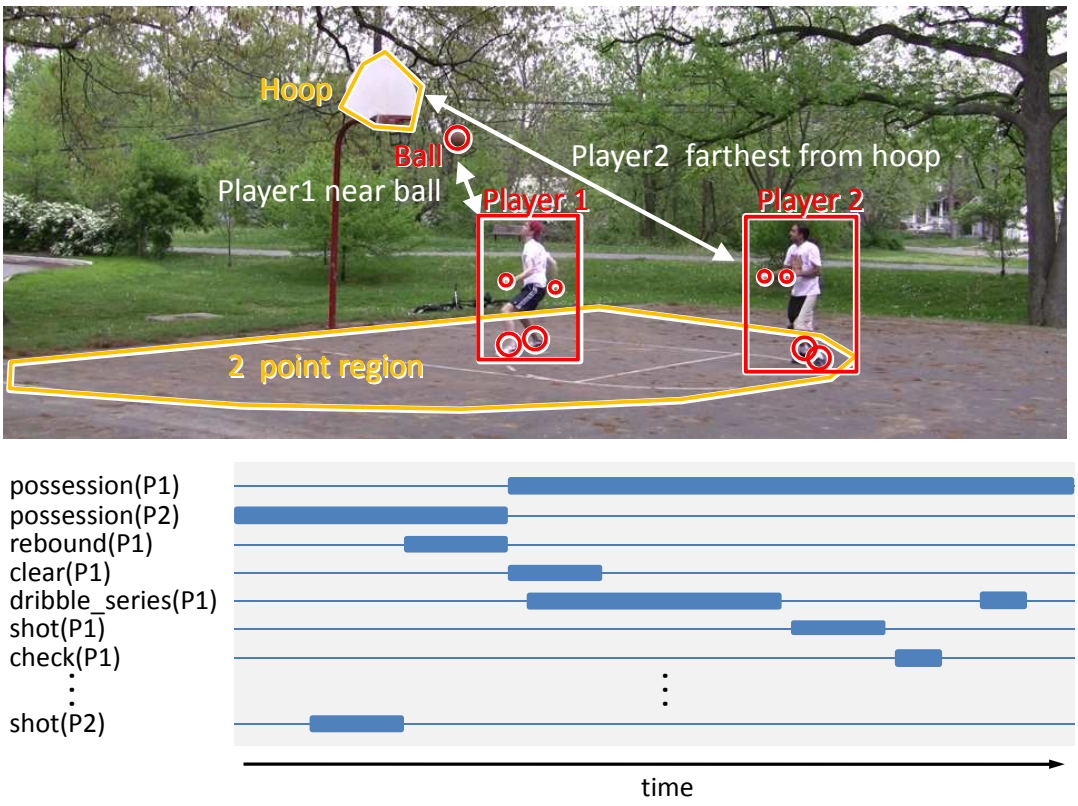


Figure 1.1: Sample structured scenario analysis task, with intermediate and final results. Intermediate results include observed trajectories of people and objects, relationships between people and objects, and relationships between people and objects to the scene (e.g., court and hoop). The final result, an annotation of high-level events and properties, can be obtained by modeling spatio-temporal relationships between events, properties, and observations.

which player is farthest from the hoop, if players are in the two point region, or if the ball is near the hoop. Finally, the framework needs to model the spatio-temporal structure of high-level events and properties, as well as observed relationships. For example, the *check* event must occur after each *shot made* or *out of bounds* events. This spatio-temporal structure can overcome observation deficiencies, so if a *shot made* event is observed because the ball appears directly under the hoop, but it is not followed by an observed *check* event, then it is possible that the shot was missed and the ball appeared to be under the hoop only due to depth ambiguity. These tasks

are not specific to the one-on-one basketball scenario and can generally facilitate the analysis of most scenarios involving interactions between people, objects, and their environment.

This work introduces the following contributions to the task of human activity understanding, each related to one of the intermediate tasks described above: 1) a framework that efficiently detects and tracks multiple interacting people and their limbs, 2) an event recognition approach that integrates both logical and probabilistic reasoning in analyzing the spatio-temporal structure of multi-agent scenarios, and 3) an effective computational model of the visibility constraints imposed on humans as they navigate through their environment.[1]

1.1 Tracking people and their limbs

First, we describe a framework that leverages mixed probabilistic and deterministic networks and their AND/OR search space to efficiently find and track the hands and feet of multiple interacting humans in 2D from a single camera view. Our framework detects and tracks *multiple* people's heads, hands, and feet through *partial* or *full occlusion*; requires few constraints (does not require multiple views, high image resolution, knowledge of performed activities, or large training sets); and makes use of constraints and AND/OR Branch-and-Bound with lazy evaluation and carefully computed bounds to efficiently solve the complex network that results from the consideration of inter-person occlusion. Our main contributions are

[1]The work on visibility has been published in [40].

1) a multi-person part-based formulation that emphasizes extremities and allows for the globally optimal solution to be obtained in each frame, and 2) an efficient and exact optimization scheme that relies on AND/OR Branch-and-Bound, lazy factor evaluation, and factor cost sensitive bottom-up bound computation.

We demonstrate our approach on three datasets: the public single person HumanEva dataset, outdoor sequences where multiple people interact in a group meeting scenario, and outdoor one-on-one basketball videos. The first dataset demonstrates that our framework achieves state-of-the-art performance in the single person setting, while the last two demonstrate robustness in the presence of partial and full occlusion and fast non-trivial motion.

1.2 Multi-agent event recognition in structured scenarios

We then present a framework for the automatic recognition of complex multi-agent events in structured settings, where structure is imposed by rules that agents must follow while performing activities. Given semantic descriptions of what generally happens (i.e., rules, meaning of relevant events), and based on video analysis, we determine the events that occurred. Applied to one-on-one basketball, our framework detects and tracks players, their hands and feet, and the ball, combining these trajectories with spatio-temporal relations to generate event observations. Knowledge about spatio-temporal structure is encoded using first-order logic using an approach based on Allen's Interval Logic, and robustness to low-level observation uncertainty is provided by Markov Logic Networks (MLN). We demonstrate our

approach on 1hr (100,000 frames) of outdoor basketball videos.

1.3 Representing visibility for human activity analysis

Visibility in architectural layouts affects human navigation, so a suitable representation of visibility context is useful in understanding human activity. Motivated by studies of spatial behavior, we use a set of features from visibility analysis to represent spatial context in the interpretation of human activity. An agent's goal, belief about the world, trajectory and visible layout are considered to be random variables that evolve with time during the agent's movement, and are modeled in a Bayesian framework. We design a search-based task in a sprite-world, and compare the results of our framework to those of human subject experiments. Our findings confirm that knowledge of spatial layout improves human interpretations of the trajectories (implying that visibility context is useful in this task). Since our framework demonstrates performance close to that of human subjects with knowledge of spatial layout, our findings confirm that our model makes adequate use of visibility context. In addition, the representation we use for visibility context allows our model to generalize well when presented with new scenes.

1.4 Organization

We begin by introducing our tracking framework in chapter 2 that produces trajectories of people and their limbs for further high-level analysis. In chapter 3, we propose a high-level reasoning approach that makes use of noisy low-level in-

put trajectories produced by our tracker to annotate events observed in multi-agent scenarios that exhibit spatio-temporal structure. In chapter 4, we describe a computational approach that captures the effects of visibility imposed on agents during spatial behavior. Finally, in chapter 5, we provide our concluding remarks.

Chapter 2

Tracking people's hands and feet using mixed network AND/OR search

2.1 Overview

The difficult problem of tracking people's hands and feet has been widely studied, as its solution is often required for higher-level reasoning about human activities. Even in the single person, known activity case, ambiguities are introduced by substantial appearance variations and possible self-occlusions. Reasoning about the hands and feet of multiple interacting people is more difficult due to inter-person occlusion, particularly without the simplifying assumption that the space of poses and movements is constrained by a set of known activities.

Our goal is to track people's hands and feet efficiently and reliably, without strong assumptions on pose or activity space. Hand and foot trajectories provide sufficient information for reasoning about many scenarios (e.g. people interacting with people, people interacting with objects, etc). We reason about inner joints (knees, elbows, neck and waist) only to ensure that the solution matches the image and satisfies physical constraints; however, we do not track them over time, to reduce computational complexity. Thus, we first detect and track people, obtain a set of extremity detections separately for each person, and extend them to tracks to

fill in periods where extremities are not detected. These tracks become extremity candidates, which are subsequently labeled as people's hands and feet.

We formulate the extremity candidate to person assignment problem using mixed probabilistic and deterministic networks. The probabilistic network is an undirected graphical model which evaluates the image likelihood given a body configuration by decomposing the overall likelihood into a product of factors. These factors measure image likelihoods of individual body parts or of pairwise relationships between them. The deterministic network enforces hard constraints (i.e., probabilities of 0 or 1) which ensure that body segments have bounded length, or that two overlapping extremity candidates can only be assigned simultaneously to two limbs if they are observed to occlude each other. To ensure temporal consistency, we add a temporal transition model between hand/foot assignments in consecutive frames.

To solve the assignment problem, we perform AND/OR search on our mixed deterministic and probabilistic network[11, 38].[1] In the presence of determinism, AND/OR search has been shown to reduce search space (and complexity) by checking hard constraints during the search process to prune inconsistent paths early. Moreover, given suitable bounds on the optimal solution, AND/OR Branch-and-Bound [36] can provide a dramatic additional reduction of the search space while

[1]Note that in [38], the authors use the term *mixed networks* to denote the case where a belief network and a constraint network are mixed. In our case, we use an undirected graphical model to represent the (non-deterministic) probability of a configuration, and use the term *mixed network* because we are combining this probabilistic graphical model with a deterministic constraint network.

still ensuring that the globally optimal solution is found. An important aspect of our approach is that it reduces image likelihood evaluations during search by computing factor entries on-demand; this requires careful computation of bounds for Branch-and-Bound, since a naive approach tends to touch all factor entries. We believe that our lazy factor evaluation approach, coupled with factor cost sensitive bounds computation, is generally applicable to other machine learning tasks where the computation of factors dominates total inference times.

The resulting framework has the following desirable properties: 1) it detects and tracks *multiple* people's heads, hands, and feet through *partial* or *full occlusion*; 2) it requires few constraints (does not require multiple views, high image resolution, knowledge of performed activities, or large training sets); and 3) it exploits determinism during AND/OR search, and reduces the number of likelihood evaluations through lazy factor evaluation and cost sensitive bound computation, while still obtaining the exact solution to the complex loopy network generated by considering inter-person occlusion.

Our main contributions are 1) a multi-person part-based formulation that emphasizes extremities and allows for the globally optimal solution to be obtained in each frame, and 2) an efficient and exact optimization scheme that relies on AND/OR Branch-and-Bound, lazy factor evaluation, and factor cost sensitive bottom-up bound computation. The first contribution is important because few approaches currently exist that deal with the difficulties of pose estimation for multiple interacting people. The novelty in the second contribution lies in our bottom-up bound computation approach that focuses on evaluating as few factor entries (i.e. image

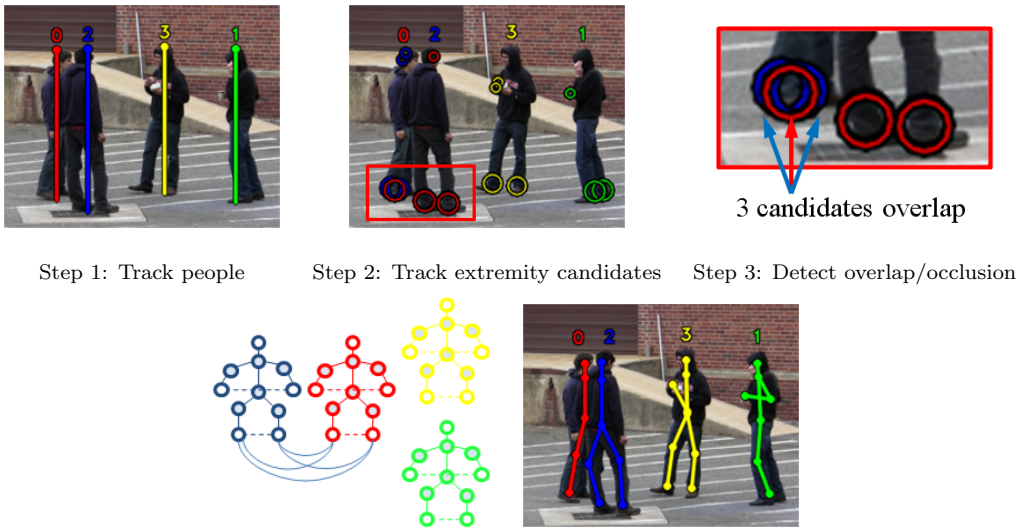


Figure 2.1: Framework overview. Assignments are also tracked temporally (not shown).

likelihoods) as possible.

2.1.1 Related Work

Several approaches to body part tracking deal with self-occlusion and appearance variations by mapping a set of high-dimensional features computed from a person's image region to low dimensional coordinates, such as joint locations, angles, or some other latent variable [33, 66]. These approaches can obtain good results, but either restrict the activity and require cyclic behavior [33], or require very large training sets [66] for activity independence. Additionally, they rely on global shape or image features which become unreliable when multiple people occlude each other.

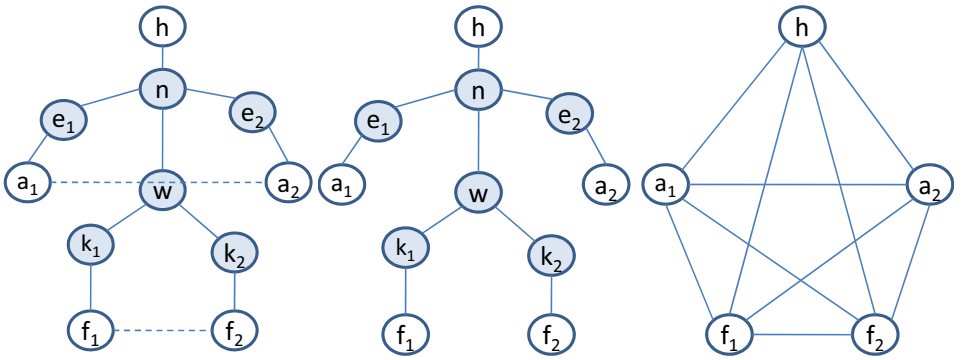
Part-based approaches reduce the reliance on large training sets and activity constraints by modeling the human body as a set of articulating body parts. Generally, these approaches detect a set of candidates for each body part and assemble

them according to an articulated model that imposes local constraints between body parts. To handle occlusion, some maintain a tree structure by applying dynamic programming in stages (detecting the first arm/leg, and then searching for a second, given the first) [15, 48]. Others use loopy graphs to directly incorporate part self-occlusion, and resort to approximate inference methods such as Belief Propagation (BP) [63, 59, 20]. Recently, Jiang and Martin [27] used a nontree model with hard mutual exclusion constraints to deal with occlusion, whose globally optimal solution is approximated by the relaxation of an integer linear program (ILP). To incorporate arbitrary pairwise constraints, Ren *et al.* [49] approximately solve an integer quadratic program (IQP) by a linear relaxation and subsequent gradient descent step. Hua *et al.* [24] employ a Belief Propagation Monte Carlo approach, which detects a small set of part candidates and uses them to construct an importance sampling distribution for non-parametric message passing.

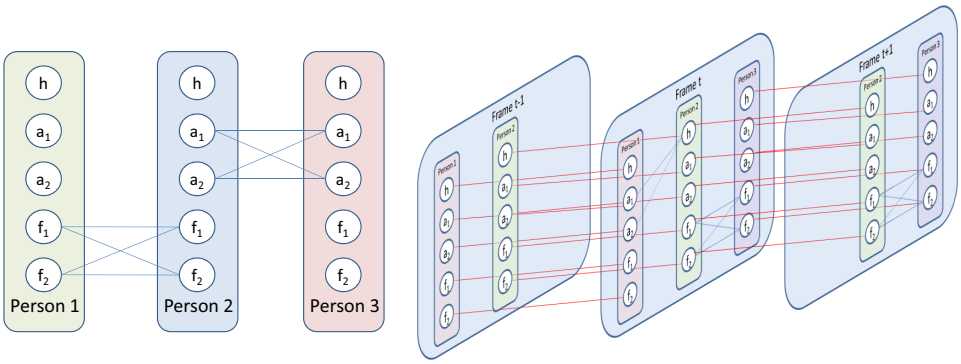
Like many of these approaches, ours uses an articulated model to factorize the overall score into a product of local part scores; however, our variables represent part endpoints, instead of part candidates. This allows us to detect only extremity endpoints (head, hands, feet) cheaply, constrain the remaining endpoints (neck, waist, knees, elbows) using length constraints, and evaluate part likelihoods on-demand (instead of searching for all parts independently over the whole image). We reason about self-occlusion and inter-person occlusion at the extremity level by dynamically adding hard constraints between extremities, and instead of resorting to approximate methods, we obtain the globally optimal solution to the resulting loopy network using AND/OR Branch-and-Bound [36].

Recent approaches to tracking multiple people simplify the problem by assuming a small set of activities, such as walking, running, etc. [2, 17, 70]. Once each person is localized, these approaches generally infer each person's pose independently, using strong motion and pose assumptions to deal with inter-person occlusion. Park and Aggarwal [43] track interacting humans and their body parts without this assumption by using a hierarchical blob-based approach; however, they deal with only two interacting people and depend on body parts to have different colors. Though we still require that people are first localized, hands and feet are assigned globally, considering inter-person occlusion constraints. The idea of dynamically adding constraints between person tracks has been explored in [69]; here, we dynamically add constraints between pairs of potentially occluding extremities. More recently, Eichner and Ferrari [13] introduce an occlusion predictor and mutual exclusion terms between body parts of different people to jointly model poses of interacting people. The model includes only upper body pose and requires approximate inference due to the resulting model complexity. In contrast, our proposed approach models both upper and lower body pose and is able to obtain the globally optimal solution in each frame. Although our approach currently employs hard mutual exclusion occlusion constraints, it can be extended to use a probabilistic occlusion model as proposed in [13].

Zhu *et al*. [72] also employed AND/OR graphs for determining human pose, but only to compactly represent the pose space learned during training. They manually create the structure of AND/OR graphs, and automatically learn parameters of potentials defined on the graph. In our work, AND/OR graphs are used for ef-



(a) Probabilistic Image likelihoods **(b)** Deterministic Length constraints **(c)** Deterministic Self-occlusion



(d) Deterministic Inter-person occlusion **(e)** Probabilistic Temporal

Figure 2.2: Probabilistic and deterministic graphical models.

ficient search of graphical models that are dynamically created in each frame; the AND/OR graph is not explicitly created, but is implicitly searched. Since computer vision problems have large search spaces and image operations are costly, Branch-and-Bound can drastically reduce computations, and has been employed for object detection and segmentation [32, 34]. One of the main advantages of Branch-and-Bound on AND/OR graphs is that we can also greatly reduce image evaluations by only computing factor entries when necessary.

2.2 Probabilistic and deterministic graphical models

In this section we describe our mixed probabilistic and deterministic graphical model that inherently handles multiple people. In order to construct our graphical model, we assume that bounding boxes have been obtained for each person, and that each person track has a set of extremity candidate tracklets. Additionally, we require a set of functions that evaluate image likelihoods given body segment configurations. Since various detection/tracking and image likelihood approaches exist, and our graphical model does not depend on the particular approach, we first describe our general model, and then provide implementation details in section 2.4.

2.2.1 Probabilistic graphical model

We use an undirected graphical model $\mathcal{P} = (X, D, F)$ to represent the image likelihood of a configuration. The nodes $X = \{X_i^p | i = h, a_1, a_2, f_1, f_2, n, e_1, e_2, w, k_1, k_2\}$, are the locations of the head (h), hands (a_1, a_2), feet (f_1, f_2), neck (n), elbows (e_1, e_2), waist (w), and knees (k_1, k_2), of each person (indexed by p). Note that each X_i^p denotes the endpoint of one or more body segments in the articulated model, e.g., $X_{k_1}^p$ and $X_{f_1}^p$ are endpoints of a lower leg segment. The discrete domain $D_i^p \in D$ of each variable, is a set of candidate locations plus the *unknown* value. Candidates for head, hands, and feet are the tracks described in section 2.4.1; candidates for the remaining *inner joints*, are obtained as described in section 2.2.4. The *unknown* value allows for missed detections, when true hands and feet are not present among the choices, and false positives, when extremity candidates

are not actually hands and feet. Finally, F is a set containing three types of factors: (1) $f_{skel}(x_i, x_j, I, \theta)$ takes two endpoints of a skeletal body segment and measures how well the segment is explained by the image (by edges, segmentation, etc); (2) $f_{app}(x_i, x_j, I, \theta)$ takes two endpoints of symmetric body parts (hands and feet), and enforces appearance similarity; and (3) $f_{unk}(x_i)$ is a constant penalty c_{unk} if x_i is unknown, and 1 otherwise. Here, I and θ are the image and external parameters (such as body segment widths), respectively. We maximize the posterior distribution, $P(X|I,\theta) \propto P(I|X,\theta)P(X|\theta)$, such that

$$P(I|X,\theta) \propto \prod_{p} \prod_{(X_i^p, X_j^p) \in E} f_{ij}(X_i^p, X_j^p, I, \theta)$$

and

$$P(X|\theta) \propto \prod_{p} \prod_{i} f_{unk}(X_i^p) \prod_{(X_i^p, X_j^p) \in E_{skel}} f_{ij}(X_i^p, X_j^p, \theta)$$

where E is the set of edges in model, $f_{ij} = f_{skel}$ if nodes i and j define a body segment, and $f_{ij} = f_{app}$ if nodes i and j are symmetric endpoints; skeletal and symmetric edges are solid and dotted in figure 2.2a, respectively. See section 2.4 for additional details on factors. Instead of precomputing the factors f_{ij} (which is costly, as we will show), we employ a lazy evaluation scheme, evaluating each factor entry the first time that it is needed during search. The prior $P(X|\theta)$ penalizes unknown locations and can include other priors on nodes, e.g., pairwise length priors.

2.2.2 Deterministic constraints

A deterministic network $\mathcal{R} = (X, D, C)$ encodes length and occlusion constraints. Variables X and their domains D are the same as in the previous section.

2.2.2.1 Length constraints

Length constraints, depicted in figure 2.2b, ensure that a body segment defined by two endpoints has bounded length. The motivation for hard length constraints is that body segment length is bounded in 3D as a ratio of height, and will be bounded even after projection to 2D under mild assumptions (i.e., camera is not pointed down toward people's heads). As long as length constraints are satisfied, we do not prefer one length over another since the same pose rotated with respect to the camera can yield segment lengths of zero due to foreshortening (so we use uniform length priors in 2.2.1). Minimum lengths can also be imposed for practical reasons, e.g., to avoid the degenerate case of zero-length segments.

2.2.2.2 Occlusion constraints

Intra- and inter-person occlusion constraints are added only between extremities (not inner joints), as we are interested mainly in tracking them. As figure 2.3 shows, two extremity tracklets can overlap either due to occlusion (first row) or due to detector false positives (second and third rows). In the former case, each tracklet corresponds to a real extremity, so both should be assigned; in the latter, only one tracklet should be assigned to a person. To be conservative, when two candidates overlap (as in figures 2.1 and 2.3), they can be assigned simultaneously only if there is visual evidence that they occlude each other; here, visual evidence for occlusion consists of observing two initially non-overlapping candidates move toward each other and partially overlap. If two candidates $v \in D_i^p$ and $v' \in D_j^q$

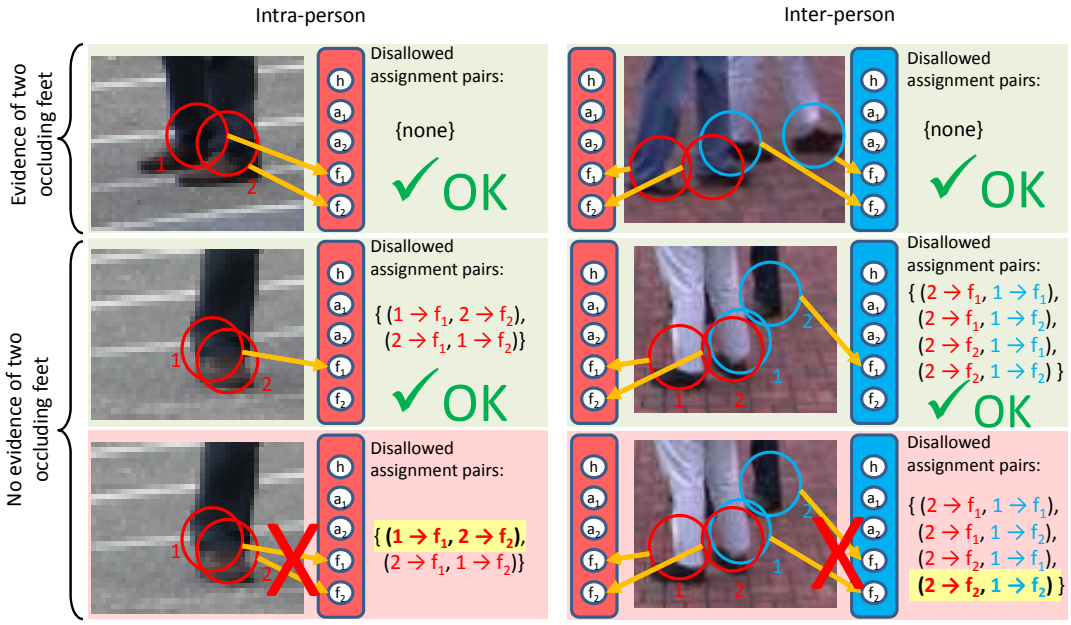


Figure 2.3: Occlusion constraints: simultaneous assignment of overlapping candidates is disallowed if there is no evidence that the overlap is caused by occlusion. When there is visual evidence of occlusion, the *disallowed assignment pairs* list is empty, and two overlapping detections can be simultaneously assigned. When there is no visual evidence of occlusion, the *disallowed assignment pairs* list ensures that only one of the overlapping detections is assigned. In this figure, disallowed pairs included by an assignment are highlighted in yellow.

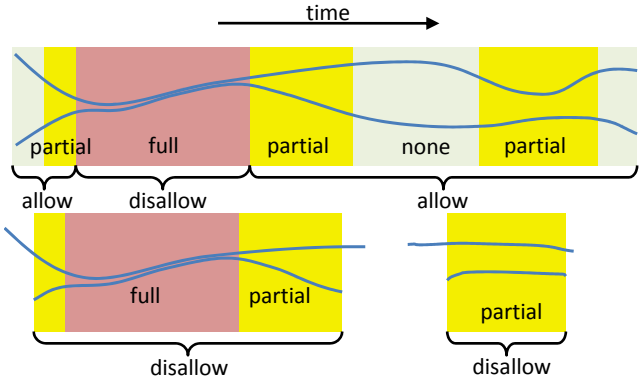


Figure 2.4: Occlusion evidence over time. Partial overlap: if a period of no overlap (between tracklet pairs) occurs immediately before or after a period of partial overlap, then this is counted as evidence of occlusion for the partial overlap period, and simultaneous assignment is allowed. Full overlap: during periods of full overlap, the track of the occluded part is assumed to be unreliable and only one of the overlapping tracks can be assigned. No overlap: simultaneous assignment is always allowed during periods of no overlap.

overlap for a period of time but no evidence of occlusion is observed, then mutual exclusion constraints are automatically added between X_i^p and X_j^q during that time period, preventing v and v' from being simultaneously assigned.

We use a two threshold approach for determining whether occlusion evidence can be used to allow simultaneous assignment of overlapping tracklets. Our approach is based on the following assumptions: 1) low-level trackers can generally track extremities through partial occlusion if they were at some point observed in isolation, and 2) low-level trackers generally fail for extremities that become fully occluded. Using these assumptions, we use two thresholds on the area of overlap between two tracked extremities to create three types of relationships between extremity pairs: full overlap, partial overlap, and no overlap. During a period of partial overlap, we allow two tracklets to be simultaneously assigned to two body parts only if there is a period of no overlap immediately before or after the partial overlap period (due to first assumption). During a period of full overlap, tracklets are not allowed to be simultaneously assigned, due to our second assumption. During periods of no overlap we allow simultaneous assignment. See figure 2.4 for an illustration of the two threshold approach.

Both intra- and inter-person occlusion are treated the same (in the former case, $p = q$). Figures 2.2c and 2.2d show intra- and inter-person constraints, respectively. Note that in the former we show the worst case scenario (all extremity pairs have occlusion constraints), but in the latter we show the case where only some pairs have overlapping domains. A graph with fully connected extremities is very unlikely, because length constraints cause domains to be spatially localized; thus,

hand candidates generally do not overlap foot candidates, and a person only has constraints between immediate neighbors.

2.2.3 Temporal assignment tracking

Temporal tracking of assignments is performed by introducing pairwise factors between corresponding extremities of people who appear in consecutive frames. Figure 2.2e shows the edges introduced by these factors. These factors, $f_{trans}(x_i^{t-1}, x_i^t)$, enforce assignment consistency between consecutive frames (see section 2.4 for the specific factors used in our experiments).

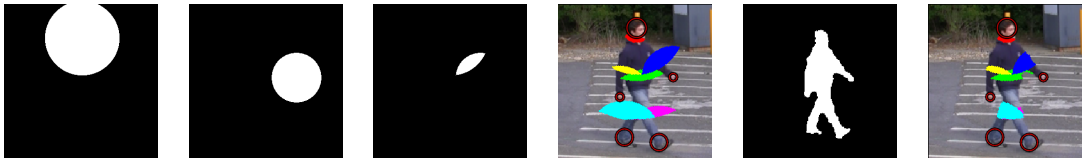
Structural changes over time can result in a complex overall graph, so we approximate the solution for a sequence as follows. First, we obtain the exact top k solutions in each frame using AND/OR Branch-and-Bound on the mixed network defined by $\mathcal{P}$ and $\mathcal{R}$ (ignoring temporal factors). The transition probability between two frames is the product of all temporal factors between extremity nodes that appear in both frames,

$$f_{trans}(X^{t-1}, X^t) = \prod_p \prod_i f_{trans}(X_i^{p,t-1}, X_i^{p,t}).$$

Given the top k solutions for each frame, we obtain the best assignment sequence using dynamic programming.

2.2.4 Node domains

Recall that for extremities, $X_e^p = \{X_i^p | i = h, a_1, a_2, f_1, f_2\}$, domains contain the extremity candidates described in section 2.4.1, plus the *unknown* state. However,



(a) (b) (c) (d) (e) (f)

Figure 2.5: Example of how inner joint domains are obtained. Fixed head and hand locations each constrain the location of the elbow, (a) and (b), respectively, and the elbow must lie in their intersection (c). The resulting domain for each inner joint (color-coded) can be obtained this way given all extremities (d). If we have a silhouette (e), we can further constrain locations (f).

we do not explicitly detect internal joints. One advantage of our formulation is that we can use the extremity candidates and hard length constraints to obtain compact regions of feasibility for each internal joint. For example, assume that we fix an extremity assignment; then, given an extremity location, any internal joint *must* lie inside of a circular region centered at that extremity, with radius equal to the maximum allowable distance imposed by the length constraints. To satisfy all constraints, an internal joint must lie in the intersection of the feasible regions given each fixed extremity. If $D_{ij}^p(X_j^p)$ denotes the feasible domain of inner joint X_i^p given extremity X_j^p, then the domain of the inner joint is $D_i^p = \bigcap_{j,\ \text{s.t.} x_j \in X_e^p} D_{ij}^p(x_j)$. Figure 2.5 depicts this process. To obtain internal joint domains without fixing any one set of extremities, we first take the union of all feasible joint regions given each possible assignment for an extremity, and then take the intersection over all extremities: $D_i^p = \bigcap_{j,\ \text{s.t.} x_j \in X_e^p} \left(\bigcup_{v \in D_j^p} D_{ij}^p(v) \right)$. In our implementation, we create the discrete domains for the inner joints by discretizing the region into a uniform grid and selecting feasible grid points as the domains.

2.3 AND/OR search with costly factors

Consider the problem as described in section 2.2.1, where the waist and neck joints, together with a width parameter, define the torso segment of the body. If there are N_w candidate locations for the waist point, and N_n locations for the neck point, then there are $N_w N_n$ candidate configurations for the torso segment. Thus, the factor that evaluates local image likelihoods of torso configurations would need to be evaluated $N_w N_n$ times to precompute all factor entries before processing. If N_w and N_n values are high enough relative to the treewidth of the graphical model (as they are in our experiments), the process of precomputing factor entries can be more time-consuming than the subsequent optimization problem. Since informed search (e.g. Branch-and-Bound) avoids a large part of the search space, lazy evaluation can be employed to avoid image evaluations. However, a straightforward computation of the upper bounds needed to guide the search (e.g., via MBE [36]) would result in the evaluation of all factor entries. By using the careful bound computation approach we describe below, AND/OR search spaces can be used to reduce evaluation cost while obtaining the exact global solution to the optimization problem. Unlike other approaches, such as Belief Propagation, whose performance degrades as determinism (probabilities close or equal to 0 or 1) is introduced, AND/OR Branch-and-Bound is able to leverage determinism present in the network.

We will first briefly summarize AND/OR search spaces [11], mixed networks [38], and AND/OR Branch-and-Bound [36]. We will then describe our proposed approach to efficiently compute upper bounds directly from the data while evaluating

few costly factor entries.

2.3.1 AND/OR search spaces

A graphical model $\mathcal{P} = (X, D, F)$ is defined by a set of variables $X = \{x_1, \ldots x_n\}$, the domain $D_i \in D$ of each variable, and a set of functions (or factors) F defined on subsets of X. Given such a graphical model, its *primal graph* is the undirected graph $G = (V, E)$ whose nodes V are the variables, X, and whose edge set E is formed by connecting any two nodes if their corresponding variables appear together as arguments in one or more of the factors in F (see Fig 2.6a). A *pseudo tree* $\mathcal{T} = (V, E')$, is a directed rooted tree defined on all of the graph nodes; any arc of G which is not included in E' is a back-arc (see Fig 2.6b). Given this *pseudo tree*, the associated AND/OR *search tree* has alternating levels of OR and AND nodes, labeled X_i and $\langle X_i, x_i \rangle$, respectively, where X_i is one of the variables in X, and x_i is a value from D_i. The root of the search tree is an OR node labeled with the root of $\mathcal{T}$. Each child of an OR node X_i, labeled $\langle X_i, x_i \rangle$, represents an instantiation of X_i with a value x_i. The children of each OR node $\langle X_i, x_i \rangle$ are the children of X_i in the pseudo tree $\mathcal{T}$. Thus, depth first traversal of the AND/OR search space begins with the root node, and alternates between choosing from possible assignments for a variable at OR nodes and decomposing the search into independent sub-problems at AND nodes. A solution is a subtree $T \subseteq \mathcal{T}$, and is defined as follows: (1) it contains the root node, (2) each OR node in T must have exactly one of its successors in T, and (3) each non-terminal AND node in T must have all of its successors in T (see

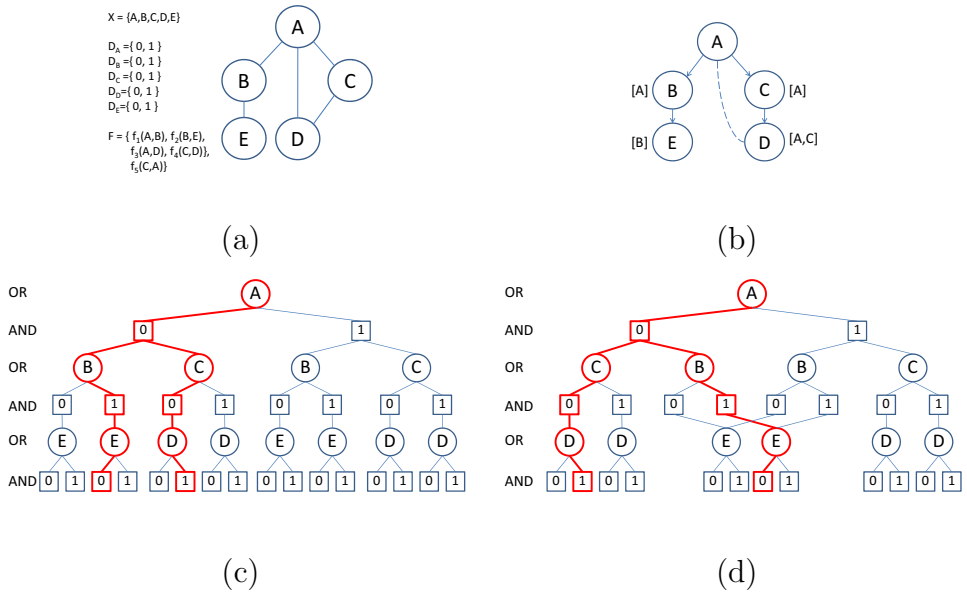


Figure 2.6: Sample AND/OR search space: (a) network description and primal graph, (b) pseudo-tree (with the parent set used for defining OR context in square brackets), (c) AND/OR search tree, with solution highlighted, (d) AND/OR search graph after merging nodes using OR context.

Fig 2.6c).

Each node X_i of the pseudo tree has an associated bucket $B_{\mathcal{T}}(X_i)$ containing each factor in F whose scope includes X_i and is fully contained along the path from the root down to X_i. During the search process, at each AND node $n = \langle X_i, x_i \rangle$, the factors in $B_{\mathcal{T}}(X_i)$ can be evaluated, as all their arguments have been instantiated along the path from the root to node n. During depth first traversal of the graph, factors in a node's bucket are evaluated when the node is first opened. After a node's subtree has been evaluated, the partial solution of that subtree is propagated to its predecessor. For a max-product task such as ours, where we maximize the product of the factors in F, the value at an OR node is the maximum of its children, and the value at each AND node is the product of the values of its children and the values of factors in its bucket $B_{\mathcal{T}}(X_i)$.

The AND/OR *search graph* can be implicitly searched by caching solutions at OR nodes. A node n is cached based on values assigned to its *parent set*, which is defined as the union of all ancestors of n in the pseudo-tree connected by an edge in the *primal graph* to n or its descendants in the pseudo-tree. The graph size and search complexity are determined by the maximum parent set size in the graph. Letting the maximum parent set size be the *induced width* (or treewidth) and denoting it by w, graph size and hence search complexity are $O(nk^w)$, where $k = \max_i |D_i|$ is the maximum domain size. Parent sets are directly affected by the ordering of nodes in the search pseudo-tree, so it is important to choose an ordering which yields the smallest possible parent sets. This can only be done efficiently by greedy approximation algorithms such as Min-Fill [36]. Figures 2.6b and 2.6d respectively show the parent sets in square brackets for each variable, and the implicit AND/OR graph that results.

2.3.2 Mixed networks

Given a constraint network $\mathcal{R} = (X, D, C)$, where X and D are defined as above, and C is a set of deterministic constraints defined on subsets of X which allow or disallow certain tuples of variable assignments, AND/OR search can be modified to check constraints of $\mathcal{R}$ while maximizing $\mathcal{P}$. This can be done by replacing the *primal graph* described above with the union of the primal graphs of $\mathcal{P}$ and $\mathcal{R}$. During the search process, constraints (and constraint processing techniques) can be applied to prune inconsistent paths early (see [38] for details). In our case, it

is beneficial to avoid exploring portions of the search tree which have no solutions by performing backtrack free search. This can be done by preprocessing constraints using Bucket Elimination (BE) [10]; in cases where BE is infeasible, Mini-bucket Elimination (MBE) [28] can be used to reduce the amount of explored dead ends.

2.3.3 Branch-and-Bound

Branch-and-Bound [36] can further reduce the search space and complexity required to obtain the optimal solution. Branch-and-Bound search involves updating the lower bound on the best solutions each time a solution sub-tree is traversed. The lower bound, coupled with upper bounds on best extensions of a partial solution can be used to prune the search space. In particular, a node is opened only if the upper bound on extensions of the current partial solution is greater than the current lower bound. Unlike the lower bound, which is computed during search, the upper bound on the value of a subtree is obtained from the graphical model before search, using the process of Mini-Bucket Elimination (MBE) [28, 36], which partitions buckets with large parent sets into smaller mini-buckets. This partitioning ensures that the approximate best solution has a score greater than or equal to that of the original best solution, but is less expensive to compute.

2.3.4 Branch-and-Bound with costly factors

A disadvantage of Mini-Bucket Elimination is that it evaluates most if not all factor entries. This is usually acceptable, but in our case accessing the value of a factor

Algorithm 1 AND/OR Branch-And-Bound with Lazy Evaluation of Costly Factors

Inputs:

(1) probabilistic and constraint networks, $\mathcal{R} = (X, D, C)$ and $\mathcal{P} = (X, D, F)$, respectively

(2) factor entry evaluation costs, α_j for each $f_j \in F$

(3) upper bound cost ratio, r

Outputs: Top k assignments $x_1, \ldots, x_k$, ranked by $\mathcal{P}$ such that $x_i \in \mathcal{R}$.

compute variable ordering and pseudo tree $\mathcal{T}$ using Min-fill heuristics
process constraints, $C' \leftarrow \mathit{process}(C, \mathcal{T})$
$F_{ub} \leftarrow F$, $\beta \leftarrow \sum_j \beta_j = \sum_j \alpha_j \prod_{X_i \in \mathit{scope}(f_j)} |D_i|$
while $\sum_j$,s.t. $_{f_j \in F_{ub}} \beta_j > r\beta$ **do**
    $f_j \leftarrow \arg\max_{f_j \in F_{ub}} \beta_j$
    $F_{ub} \leftarrow F_{ub} \setminus f_j$
end while
compute upper bounds, $U \leftarrow \mathit{bounds}(C', F_{ub})$, by MBE or AND/OR traversal
perform AND/OR Branch-and-Bound on $\mathcal{R}$ and $\mathcal{P}$ using constraints C' and bounds U
return top k assignments

entry is an expensive operation. If the cost of evaluating one entry of factor $f_j \in F$ is α_j for $j = 1, \ldots, m$, then the cost of evaluating all entries of f_j is $\beta_j = \alpha_j \prod_{X_i \in \mathit{scope}(f_j)} |D_i|$. To avoid evaluating all entries, we first sort factors by β_j, and then iteratively remove the largest factors until the total remaining evaluation cost is below some ratio r. To ensure that any solution of the resulting graphical model bounds the original from above, we must replace each removed factor by a constant that bounds all of its entries. By construction, the factors described hereafter are always bounded by 1, so upper bounds obtained by MBE on the reduced problem can be used to obtain the exact global solution, while ensuring that few entries need to be evaluated in the process.

Our lazy evaluation approach to dealing with costly factors is summarized by

Algorithm 1. The first step is to compute the variable ordering which is used to construct the pseudo tree $\mathcal{T}$. Constraint processing is then performed to yield a new set of constraint factors C'. Note that in our case, constraint processing yields a new set of constraints that are consistent with the initial pseudo tree $\mathcal{T}$. In the next steps, a subset of factors $F_{ub} \subseteq F$ is chosen for upper bound computation to ensure that most factor entries are not evaluated in computing the upper bounds U. Finally, AND/OR Branch-and-Bound is performed on the mixed network, using the processed constraints C' and the upper bound U. The constraint processing step may be redundant if it is performed using MBE and upper bounds are also computed by MBE using the same mini-buckets. However, it is not redundant if upper bounds are computed by AND/OR traversal, as constraint processing will allow AND/OR traversal to encounter dead ends earlier. Similarly, it is not redundant if MBE is optimized to the binary nature of constraints to reduce memory and computational requirements, enabling MBE to perform less approximation (by using larger mini-buckets). Also, note that upper bounds, which are computed using MBE in [36] can also be computed by traversal of the AND/OR search space of the reduced problem. The traversal is more computationally intensive than MBE, but touches fewer of the entries in factors from F_{ub}, leading to an overall gain in efficiency.

2.4 Implementation details

In the previous sections, we have described our general formulation for a multi-person part-based hand and foot tracker, as well as an AND/OR Branch-and-Bound

approach for obtaining the exact solution efficiently while reducing image likelihood evaluations via lazy evaluation. Because the novelty of our work does not lie in low-level feature design for individual body part or body segment detection and tracking, we left these details out of the general formulation. Various approaches can be used to detect person, hand, and foot candidates [24, 2, 43, 22, 52, 55], or link them into tracklets [22, 25, 14, 35]. Similarly, various approaches exist for evaluating body segment image likelihoods [48, 15, 52, 58]. For simplicity and to demonstrate the effectiveness of our approach, we use silhouette-based approaches that require little training or parameter tuning; we base our approach on [43, 22] for person and extremity detection, [25, 14] for tracking, and [15] for image likelihood evaluation.

2.4.1 Candidate tracklets

Before we construct the mixed network for assigning extremity candidates to people, we must first track people and obtain a set of extremity candidate tracklets for each person. We take the common approach of initially tracking the bounding boxes of people, and then tracking smaller body parts ([17, 22]). Videos are preprocessed by first computing optical flow [42] and then detecting moving foreground pixels using background subtraction [30]. Once foreground pixels are obtained, we detect potential head tracklets using a data association approach based on [25]. Head tracks provide us with an estimated height, which in turn allows us to search for potential hands and feet at the appropriate scale for each person. These hand and foot detections are also linked into tracklets in a way similar to the heads. We first

describe the detection association approach, assuming that a set of detections is provided. We then provide a brief overview of the detection and post-processing steps performed for each part of interest.

2.4.1.1 Tracking by association

The algorithm presented in [25] consists of three stages, each of which involves linking the tail of one tracklet (a sequence of already linked detections) to the head of another tracklet. Our scenes have no static occluders and entries/exits are not of interest; consequently, we use a simplified approach, iteratively applying the Hungarian algorithm and considering longer time gaps between tracklets at each iteration. We avoid using motion based features that assume constant velocity since their performance suffers when people change directions often (as they would in a basketball game); we instead use optical flow based features for linking probabilities. We model each detection as a tuple $D = (c, R, X)$, containing the center c, a neighborhood R around c, and a set of pairs $X = \{(x_i, d_i)\}$. Here, x_i is a 2D image location inside R at distance d_i from the center c. To compare two detections D_1 and D_2 in frames t_1 and t_2, respectively, we propagate the x_i locations of the earlier detection D_1 using flow as $x_i^t \leftarrow x_i^{t-1} + u_{t-1,t}(x_i^{t-1})$, until we reach frame t_2, where $u_{t-1,t}(x_i^{t-1})$ is the flow vector at location x_i^{t-1} from frame $t-1$ to frame t. The linking probability P_{link} is given by

$$P_{link}(D_1, D_2) \propto \frac{1}{N_{1,2}} \sum_{x_i^{t_2} \in R_2} \exp\left(-\frac{(d_i - d(x_i^{t_2}, c_2))^2}{2\sigma^2}\right).$$

The sum is over all pixels from D_1 propagated to frame t_2 whose propagated locations $x_i^{t_2}$ are inside region R_2, and $N_{1,2}$ is the size of the union of the propagated pixels from D_1 and the pixels of D_2. The intuition behind this measure is that if D_1 and D_2 are the same detection, when propagated from t_1 to t_2, pixels inside R_1 will move to R_2 and their distance from the center will remain the same, leading to a value of 1 for the above equation. The exponential term penalizes pixels that change distance from the detection center while providing some rotational invariance.

2.4.1.2 Person tracklets

Person detections are obtained from silhouette contour peaks, similar to [22], though we do not perform silhouette-based tracking. The neighborhood region R used for association consists of pixels whose pathlength (or inner distance) from the contour head point through the foreground mask is less than $.15h$, where h is the person's height approximated from the contour points below the head. Tracklets are formed iteratively by flow-based linking with tracklet gaps of $\{1, 8, 16, 40\}$ frames. Gaps in tracklets are filled by the mean location of the pixels propagated from region R_1 of one tracklet's tail to region R_2 of the other tracklet's head. To handle longer gaps, we perform another iteration allowing gaps up to 160 frames, but flow becomes unreliable for long gaps, so the linking probability is based on the χ^2 distance measure between the two RGB histograms accumulated for each tracklet. We fill gaps between linked tracklets by meanshift tracking [8], keeping approximated locations only if meanshift tracking reaches one tracklet from the other (either forward or

backward in time).

2.4.1.3 Extremity tracklets

Hands and feet are represented as circular regions at high curvature locations on silhouette or skin blob contours (for hands only), within some distance from the head. The centers are near the wrist and ankle, with radii given as a fixed fraction of the person height ($r_{hand} = \frac{h}{30}$, $r_{foot} = \frac{h}{15}$). Curvature is approximated by the angle between the two segments formed by a contour point and its two neighbors of equal distance δ forward and backward along the contour. After non-maximal suppression, high curvature points are binned into hand and foot detections based on distance from the head, and centers are set to the centroid of pixels within distance $2r$ of the contour point, where r is the hand/foot radius. Detections are linked into tracklets using maximum gaps of $\{1, 8, 16, 40\}$ frames, filling in gaps using pixels from region R_1 of the tail propagated to region R_2 of the head.

2.4.2 Image likelihoods and priors

2.4.2.1 Body segment likelihood

Our body segment likelihoods are based on those described by Felzenszwalb *et al.* [15]. Given two end-points of a body segment and a width parameter, we can define a rectangular region representing that body segment that roughly represents the body part and its nearby neighborhood. The rectangular region is divided into multiple sub-regions R_r ([15] used two sub-regions). The likelihood of a body part

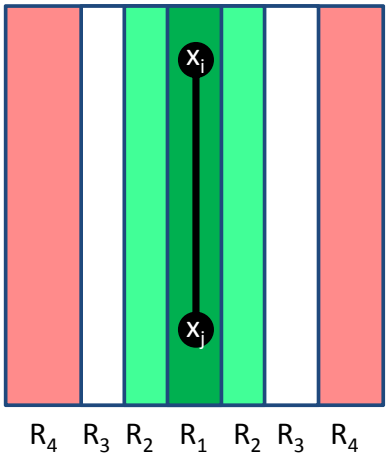


Figure 2.7: Body segment likelihoods are split into four regions, defined by their distance from the body segment.

given the two end-points is then

$$f_{skel}(x_i, x_j, I, \theta) = \prod_r q_r^{c_r}(1 - q_r)^{(a_r - c_r)},$$

where q_r is the foreground pixel probability for region R_r, a_r is the area of region R_r, and c_r is the foreground pixel count in region R_r. We use up to four regions for all body segments, defined by the distance from the center axis of a body segment, with manually selected parameters $q_1 = .99$, $q_2 = .9$, $q_3 = .5$, and $q_4 = .3$ (ordered from center-most to outer-most regions; see figure 2.7); for upper arm, lower arm, and head segments, which are smaller, we use only regions R_2 and R_4.

2.4.2.2 Appearance similarity

Appearance similarity factors represented by dotted lines in figure 2.2a are computed as $f_{app}(x_i, x_j, I, \theta) = exp(-c_{app} d_{KL})$ where $d_{KL}(x_i, x_j)$ is the symmetric Kullback-Leibler divergence between appearance models of x_i and x_j. Appearance models are represented by points sampled around the extremity locations, and divergence

is computed efficiently using Kernel Density Estimation and a fast Gaussian summation approach [41].

2.4.2.3 Temporal motion consistency

The temporal consistency factors are defined based on the tracklet linking probability P_{link} described above, modified to consider transitions to and from the *unknown* state and to allow for more flexibility for cases where flow fails (e.g., very fast motions). For the case where both x_i^{t-1} and x_i^t are actual candidates (i.e., not *unknown*) the factor is defined as

$$f_{trans}(x_i^{t-1}, x_i^t) = \max\left(c_{switch}, \frac{N_{lost}}{N_{1,2}} e^{-\frac{||c_i^{t-1} - c_i^t||^2}{2\sigma^2}} + P_{link}(x_i^{t-1}, x_i^t)\right)$$

where c_i^t is the center of x_i at time t, and N_{lost} is the number of pixels for which flow could not estimated (in [42], this occurs when forward and backward flow do not match, usually due to fast motion or occlusion). We truncate the factor to have a minimum value of c_{switch} to make large jumps equally likely. The switch from a known candidate to the *unknown* state or vice versa is given a cost of $\sqrt{c_{switch}}$, so a large jump can be seen as a switch from the tracklet of x_i^{t-1} to *unknown* followed by a switch to the tracklet of x_i^t.

2.4.2.4 Priors

Our approach does not use strong pose priors, to avoid becoming pose or activity specific. In particular, we assume a uniform distribution on body segment length

(as a proportion of height) as long as hard length constraints are satisfied. We do, however, impose simple priors on the torso and head. The torso prior penalizes torso segments that deviate much from vertical position (the types of people we can detect and track are generally close to standing position). The head prior penalizes acute neck angles to coincide with physical constraints; this prior involves three joints (head, neck, and waist), but since we have a single head location in our approach, the prior is effectively a pair-wise prior on the neck and waist variables.

2.5 Experiments

We demonstrate our approach quantitatively on three datasets: outdoor scenes containing a group of three to five interacting people; one-on-one basketball in outdoor scenes; and part of the publicly available HumanEva I dataset [60], which contains single person sequences. While our focus is on tracking extremities of *multiple* people efficiently, we use the single person HumanEva I dataset to show that our model performs comparably to state-of-the-art approaches, observing only a small performance drop in the absence of large training sets or activity models. The following subsections define our performance measures, describe dataset details, evaluate inference approach computational cost, and compare our occlusion reasoning framework to alternatives. Table 2.1 provides dataset statistics and figure 2.8 shows sample qualitative results.

2.5.1 Performance measures

We measure system performance in two ways: average pixel error and precision-recall measures. Both measures require that detected people and their extremities are associated to ground truth people and extremities. We do this hierarchically, first fixing associations between detected and ground truth people, and then computing extremity associations. A one-to-one matching between detected person bounding boxes and ground truth person tracks is chosen such that it minimizes average distance between head locations. Because our framework does not differentiate between left and right hands or feet, average pixel errors are computed from the maximum matching between detected and ground truth extremities that minimizes average error. The average pixel errors for each dataset are visually displayed in figure 2.9. Precision and recall is computed in a similar way, but we also allow detected hands (or feet) of a detected person to match the ground truth hands (or feet) of other ground truth people in addition to those belonging to the ground truth person associated at the bounding box level. This approach is more strict since a detected hand (or foot) that matches the ground truth hand (or foot) of a wrong ground truth person (according to the bounding box matching) will be counted as a false positive. A detected hand or foot is considered a false positive if the distance to its matching ground truth location is greater than one-tenth of the ground truth person height. If a detected person does not match a ground truth person, all detected extremities will be counted as false positives; similarly, if a ground truth person is not matched to a detected person, all ground truth extremities will be

Table 2.1: Datasets

	Frames	Mean±std Height	Annot. People	People/ frame
HumanEva I	5533	292 ± 34	4029	1
Group	1429	128 ± 14	5497	3-5
Basketball	101933	133 ± 27	2499	2

Table 2.2: Average pixel error on HumanEva I

	Overall	Hands	Feet
Avg over all errors	**12.65**	13.64	11.66
Fully and part. vis. only	**10.86**	11.17	10.55
Fully visible only	**10.28**	10.39	10.18

Table 2.3: Average pixel error on group dataset

	Overall	Hands	Feet
Avg err (pix)	**2.87**	**2.34**	**3.17**
Unknown, fully visible	8.0%	10.2%	5.9%
Unknown, part. visible	7.1%	10.0%	4.2%
Unknown, not visible	23.4%	35.6%	11.4%

Table 2.4: Average pixel error on basketball dataset

	Overall	Hands	Feet
Avg err (pix)	**7.42**	**8.56**	**6.68**
Unknown, fully visible	15.5%	20.6%	10.4%
Unknown, part. visible	6.4%	7.6%	5.2%
Unknown, not visible	11.2%	19.0%	3.5%

counted as false negatives. In addition, we report three types of recall measures. The first measure penalizes all missed hands/feet, counting them as false negatives regardless of their visibility (Rnv); the second discards false negatives if the missed ground truth extremity is marked not visible (Rpv); finally, the third counts only false negatives if the ground truth extremity is marked fully visible (Rv). The F1 measure (harmonic mean between precision and recall) is computed for each pair of precision and recall values.

Table 2.5: Precision-recall evaluation of candidate tracklets, single- and multi-frame assignments

	hands						
	P	Rnv	Rpv	Rv	F1nv	F1pv	F1v
HumanEva I							
candidates	0.69	0.75	0.91	0.96	0.72	0.78	0.80
assign., single-frame	0.96	0.74	0.90	0.95	0.84	0.93	0.96
assign., multi-frame	0.96	0.74	0.90	0.95	0.84	0.93	0.96
group							
candidates	0.51	0.42	0.67	0.80	0.46	0.58	0.62
assign., single-frame	0.90	0.40	0.64	0.77	0.56	0.75	0.83
assign., multi-frame	0.92	0.41	0.65	0.79	0.57	0.76	0.85
basketball,automatic							
candidates	0.51	0.36	0.46	0.53	0.42	0.49	0.52
assign., single-frame	0.74	0.33	0.43	0.50	0.46	0.55	0.59
assign., multi-frame	0.74	0.34	0.44	0.50	0.46	0.55	0.60
basketball,interactive							
candidates	0.58	0.43	0.55	0.63	0.50	0.57	0.61
assign., single-frame	0.80	0.42	0.53	0.61	0.55	0.64	0.69
assign., multi-frame	0.81	0.42	0.54	0.61	0.55	0.65	0.70
	feet						
	P	Rnv	Rpv	Rv	F1nv	F1pv	F1v
HumanEva I							
candidates	1.00	0.77	0.91	0.95	0.87	0.95	0.97
assign., single-frame	1.00	0.76	0.90	0.94	0.86	0.95	0.97
assign., multi-frame	1.00	0.76	0.90	0.94	0.86	0.94	0.97
group							
candidates	0.49	0.80	0.88	0.93	0.61	0.63	0.64
assign., single-frame	0.97	0.76	0.85	0.90	0.85	0.91	0.93
assign., multi-frame	0.98	0.77	0.85	0.90	0.86	0.91	0.94
basketball,automatic							
candidates	0.68	0.68	0.72	0.78	0.68	0.70	0.73
assign., single-frame	0.85	0.62	0.65	0.71	0.72	0.74	0.77
assign., multi-frame	0.85	0.62	0.65	0.71	0.72	0.74	0.78
basketball,interactive							
candidates	0.75	0.76	0.80	0.86	0.76	0.77	0.80
assign., single-frame	0.89	0.71	0.74	0.81	0.79	0.81	0.85
assign., multi-frame	0.90	0.72	0.75	0.82	0.80	0.82	0.86

Figure 2.8: Sample results for datasets with multiple people. Heads, hands, and feet are indicated by large, small, and medium sized circles, respectively. The radius is a fixed proportion of person height determined by the type of extremity. Color indicates the person to which each extremity belongs.

Figure 2.9: Average errors from tables 2.2, 2.3, and 2.4, drawn in context.

2.5.2 Datasets

2.5.2.1 HumanEva I dataset

We processed all frames of S2_Gestures_1, S2_Box_1, S2_Walking_1, S3_Gestures_1, S3_Box_1, and S3_Walking_1 (700 to 1100 frames per sequence), all from camera 2. The video resolution is 640 by 480 and people are on average 292 pixels tall. Errors are computed from the ground truth motion capture data projected onto the single camera image. To compare to previously reported average error measurements, we automatically filled in unknown hand and foot locations as a post-processing step using one of two approaches: for short periods where hands/feet are not known, we interpolate between known locations using cubic spline interpolation; for long periods, we set the location to a standard position for the unknown part along the vertical axis of the person's bounding box ($.5h$ and $2r_{foot}$ from the bottom of bounding box for hands and feet, respectively). Table 2.2 reports the results on the videos in their original size of 640 x 480. The errors are competitive with the state-of-the-art techniques tabulated by Martinez *et al.* in [37], where the authors themselves report average error rates of 13.2 pixels using bipedal motion constraints for the legs. The best results were obtained by [33] which were estimated by Martinez *et al.* from

the 3D errors to be about 5 to 7 pixels in 2D. This approach was activity specific and assumed cyclic motion. Other approaches had errors between 10 and 14 pixels [45, 23], and all were activity (and sometimes even view) specific. Our approach was not trained on the HumanEva I dataset, nor on any activity or view, but was still able to obtain comparable results (average errors for legs are 11.66 pixels). Since our algorithm does not train on poses and is not activity specific, we do not expect it to accurately hallucinate positions of occluded extremities. To evaluate the penalty incurred by guessing occluded extremity locations, we augmented the HumanEva I ground truth by labeling hands and feet with one of three labels: fully visible, partially visible, and not visible (fully occluded). Table 2.2 shows average errors for all ground truth extremities including fully occluded ones (first row), for fully and partially visible ground truth extremities (second row), and for fully visible ground truth extremities (third row). As expected, our system performs best when the actual body parts are not occluded, but the error introduced by interpolation is relatively small.

2.5.2.2 Outdoor group dataset

The outdoor dataset of multiple interacting people includes actions such as hand-shakes, drinking from mugs, and gestures, and contains periods of partial and full inter-person occlusion. The video resolution is 480 by 270 at a frame-rate of 30fps, with an average of 128 pixels of vertical resolution for each person. In most of the sequences, people wore similar clothing, making the task particularly difficult

in the presence of inter-person occlusion. Based on 1324 frames annotated with ground truth (5497 pose instances), the detected locations had an average error of approximately 3 pixels, as shown in table 2.3. For these videos, we do not interpolate or guess unknown hand/foot locations, but we instead report how often (as a percentage) parts are declared unknown by our system. As table 2.3 indicates, most unknown extremities are missed because they were fully occluded. Hands are occluded more often than feet (e.g., hands in pockets, hands are occluded by torso), so they are declared unknown more often than feet. Also, since they are much smaller, they are missed more often even when they are visible.

2.5.2.3 Outdoor one-on-one basketball dataset

The one-on-one basketball dataset contains videos of roughly 100,000 frames of 960 by 540 video at 30fps (about 1hr). These videos contain a large variety of natural poses that occur during a game, and include rapid motions and severe occlusion as the defensive player often maintains close proximity to the offensive player. Players are on average 132 pixels tall. Hands are more difficult to detect in these videos due to their small size and relatively fast motions. Because some of the sequences are longer than 25,000 frames, automatic head tracking fails a number of times. For this reason, we report results with both fully automatic and partially annotated bounding box tracking. Partially annotated results are obtained by allowing the user to provide input during the person bounding box tracking step to remove player track merges/switches. This does not require the user

to add missing detections, i.e., person bounding boxes are automatically detected by the person detector, and manual interaction involves only the association of a player identity to these detections to correct tracker mistakes. The amount of user interaction is minimal: for the seven sequences of roughly 100,000 frames, a total of 168 tracklets were obtained using the fully automatic approach, whereas manual intervention ensures that only 14 tracks are obtained. For evaluation, every 100^{th} frame of each sequence is manually annotated with ground truth extremity locations. Hand and foot average errors (with user interaction) are shown in table 2.4; the performance hit caused by incorrectly associating tracklets is small, as can be seen in table 2.5.

2.5.3 Inference approach

We evaluate our lazy evaluation inference approach by performing inference using (Loopy) Belief Propagation (BP), Variable (or Bucket) Elimination (VE) [10], and AND/OR Branch-and-Bound (AO). For BP, we implemented a C/C++ version of the code provided by [68]; we also used the open source library libDAI [39] and obtained similar results. We also evaluate various ways in which hard constraints are handled before evaluating probabilistic factor entries. Constraints are either ignored (only in the BP case), partially considered (P), or fully processed (F). In the first case, constraints are ignored by evaluating all factor entries during a preprocessing step that populates all factor tables with entries. In the second case, they are partially considered by first checking any immediate constraints before evaluating

a probabilistic factor (by *immediate*, we mean any constraint functions defined on the same variable subset as the factor in question, and in the AND/OR case any other constraints instantiated along the search). Finally, in the third case, they are fully processed by some constraint processing approach as described in section 2.3. For AND/OR search, we compare three approaches to dealing with upper bounds, which are computed before search from the factors (lower bounds are obtained dynamically during search). The straightforward approach is to assume a constant upper bound (C) based on knowledge of how factors are constructed; in our case no factor has a value greater than 1, so we can assume this upper bound without actually evaluating any factor entries. We also compare Mini-bucket Elimination (MBE) [28, 36] to a full traversal of the reduced AND/OR search space (AO); in both of these cases, the most costly factors are removed before bounds computation as determined by the parameter r. Thus the methods that we compare are **(BP)**, **(BP,P)**, **(VE,P)**, **(AO,P,C)**, **(AO,P,MBE)**, **(AO,F,C)**, and **(AO,F,AO)**. The naming pattern uses **(method,constraint,bounds)** triplets to describe the general inference, constraint handling, and upper bound approaches, respectively. We omit the constraint and bounds parts of the triplets when they are ignored. Also, note that because our full constraint processing consists of performing MBE on the constraints only (a bottom-up approach, with respect to the pseudo tree), **(AO,P,MBE)** and **(AO,F,MBE)** are equivalent to each other; however, other constraint processing techniques (such as propagating constraints top-down with respect to the pseudo tree) may be appropriate for the case where upper bounds are computed with MBE. Note that all of the inference approaches listed above except for **BP** and **BP,P** ob-

tain the *exact globally optimal* solution to the optimization problem in each frame. Because we are dealing with loopy graphical models, Belief Propagation is not guaranteed to converge, and the determinism (zeros and ones as probabilities) in our problems further reduces the convergence rate. In our experiments, the BP-based approaches often did not converge, leading to solutions which did not satisfy the hard constraints or attain the globally optimal solution. The reason that exact approaches work in our case is because we apply occlusion constraints at the extremity level, not between internal nodes, sufficiently reducing the overall treewidth to allow for exact inference. We report our inference approach experiments using only the group dataset, as it contains the most complex interactions.

2.5.3.1 Inference vs. evaluation

Figure 2.10 shows the average time spent evaluating image likelihoods (red) and average time performing inference (blue) for each method. For our problem, it is evident that most of the time spent assigning hands/feet is spent on evaluating image likelihoods relative to the time spent performing inference to find the best assignment. Because the inference time is so small relative to factor evaluation time, it is less useful to speed up inference itself, and more useful to avoid image likelihood evaluations by consulting constraints before evaluation and by performing lazy evaluation. Figures 2.11 and 2.12 show the total number of evaluated factor entries and speedup, respectively, as problem size varies (measured by total possible image evaluations). Two results become evident: (1) methods that leverage constraints

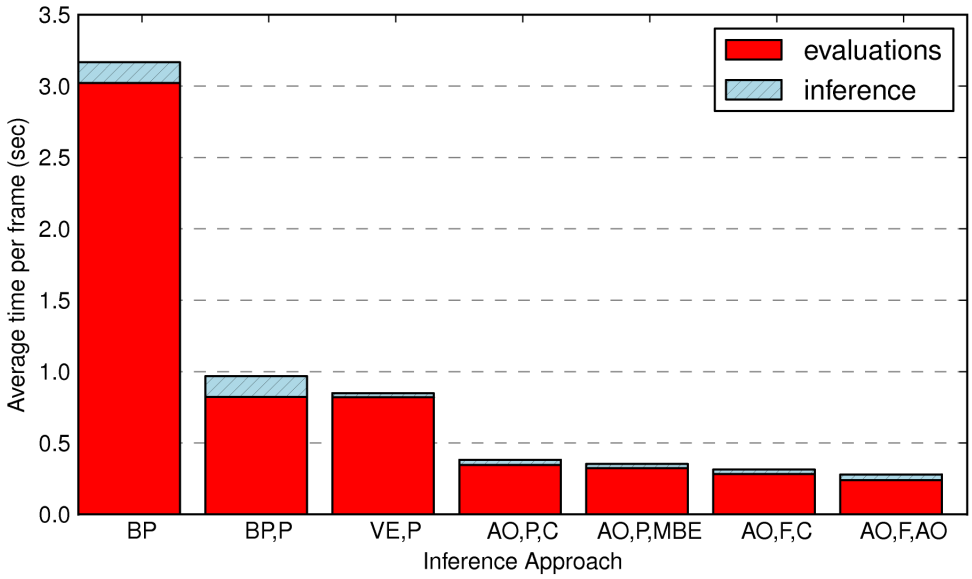


Figure 2.10: Inference and likelihood evaluation times: average time per frame by inference approach (outdoor group dataset only).

and perform lazy evaluation are faster than those that do not, and (2) methods that leverage constraints and perform lazy evaluation are more scalable than those that do not. The scalability observation is based on the fact that the percent of evaluated factor entries becomes almost constant once a certain problem size is reached (approx 300,000 total factor entries). As a result, for the largest problems, our proposed approach **AO,F,AO** is almost 25 times faster than straightforward **BP**!

2.5.3.2 Constraints

The time difference between **BP** and **BP,P** times in figure 2.10 shows us that even a simple application of constraints (pairwise constraints in this case) before

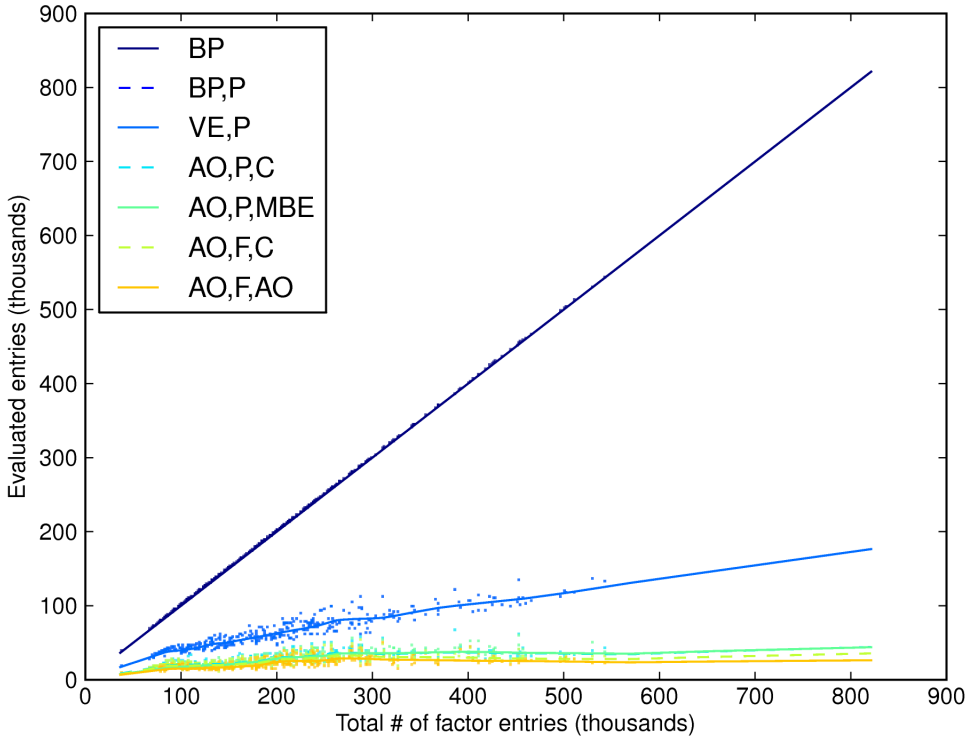


Figure 2.11: Benefits of lazy evaluation: number of image likelihoods evaluated vs number of total possible image likelihoods. The Lowess curve [7] is fitted to each method's scatter plot to show general trend (sub-sampled data-points are also shown). The curves appear in the same vertical order as in the legend, and (BP,P) and (VE,P) evaluate the same number of likelihoods, so their curves are identical.

evaluating image likelihoods can significantly reduce average times. In the absence of hard constraints, average times would be similar to **BP**. In addition, we see that performing additional constraint processing results in lower total times, even though more time is spent during inference on processing constraints, since the two approaches which use full constraint processing result in the fastest average total times.

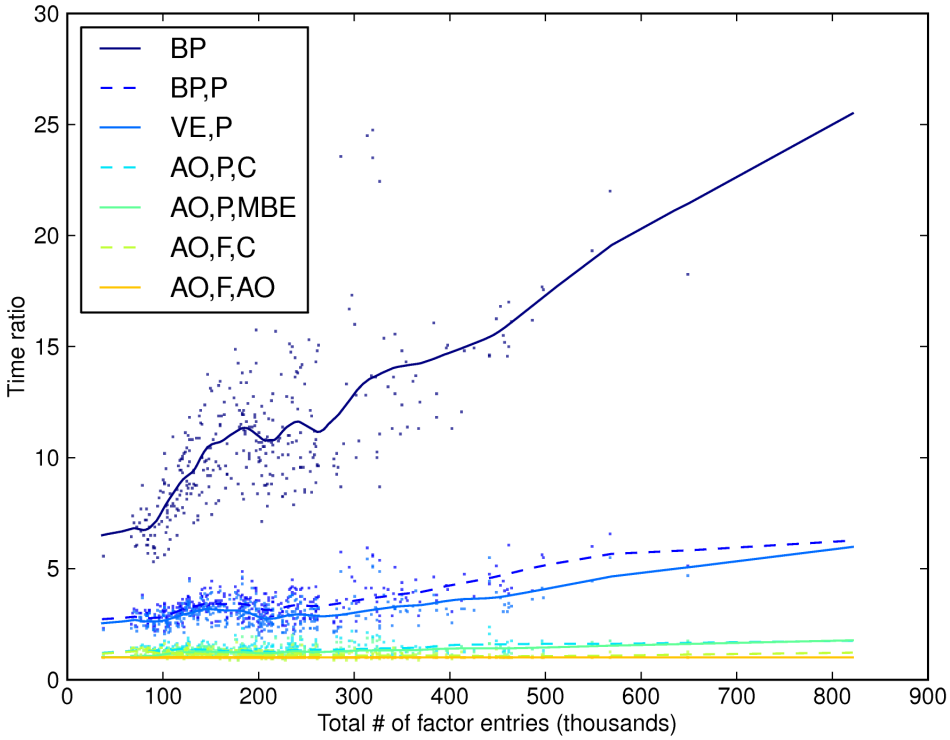


Figure 2.12: Inference speedup: time ratio of each approach to our proposed inference approach (**AO,F,AO**) by problem size. The Lowess curve [7] is fitted to each method's scatter plot to show general trend (sub-sampled data-points are also shown). The curves appear in the same vertical order as in the legend.

2.5.3.3 Upper bounds

Figures 2.10, 2.11 and 2.12, show that our approach of using full traversal of the reduced AND/OR search space for computing upper bounds results in the fastest average times. This is faster than using MBE for upper bound computation because MBE evaluates more of the entries of factors kept for upper bound computation, whereas much of the search space is pruned by the top-down search space traversal, especially when bottom-up constraint processing is performed by MBE. A surprising result is that using constant upper bounds is faster than using MBE upper bounds;

in this case, the time saved by not evaluating any entries to compute upper bounds was greater than the time saved by having more informative upper bounds as given by MBE.

2.5.3.4 Parameters

Finally, we evaluate the performance of our approach as parameters k and r are varied. Figure 2.13 shows that there is a trade-off between the quality of the upper bounds and number of evaluated entries to give better upper bounds. For high values of r (meaning more factors are kept for upper bound computation), upper bounds are more accurate, but too much time is spent evaluating upper bounds. If r decreases too much, upper bounds become uninformative, decreasing overall performance. The optimal value of r is around .10; a desirable result from the graph of time vs r is that total time does not change much for deviations from the optimal value of r. The second graph, shows total time as k, the number of solutions per connected component in the primal graph, is varied. From this graph, we see that the total time scales well with larger values of k.

2.5.4 Inter-person occlusion

We also performed experiments that compare our joint inter-person occlusion reasoning approach with two alternative approaches: (1) computing the best assignment for each person *individually* by ignoring assignments of other people, and (2) computing the best joint assignment *iteratively* by fixing the assignment of the best

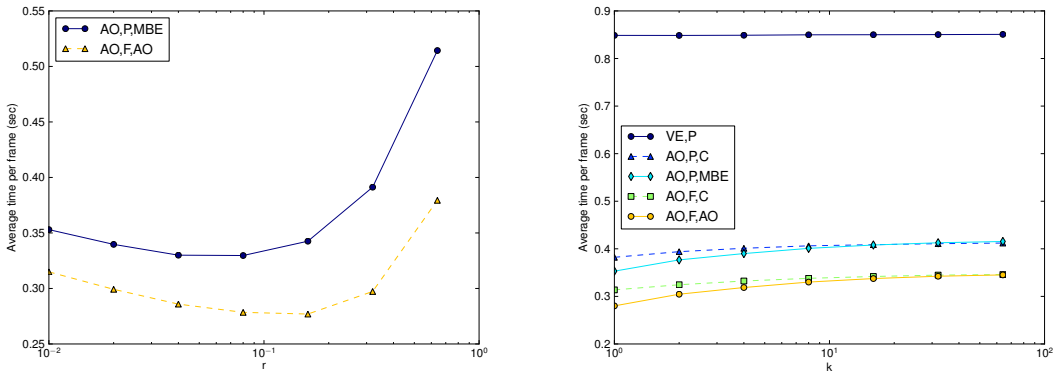


Figure 2.13: Inference parameters: average time per frame by varying k (number of top solutions) and r (ratio for upper bound computation).

scoring person that is consistent with the already-fixed set. Also, since our current implementation obtains the head location from people's bounding boxes and does not allow it to change during the assignment process, we can enforce occlusion constraints between the head and other body parts as a pre-processing step by removing from consideration any assignments that violate a head-limb constraint. Table 2.6 shows hand and foot detection results on the group dataset for various occlusion approaches, in terms of precision, recall (computed only when extremities are at least partially visible), and F1 measures. The first row shows the result of finding the best assignment for each person individually, without head constraint pre-processing (thus, skin blobs corresponding to people's faces are incorrectly assigned as hands). The second and third rows show the individual and iterative approaches, both with head constraint pre-processing. Finally, the last two rows show the results of our approach, with and without temporal assignment tracking. Note that the single-frame results are single-frame only in the sense that we are reporting the best solution found per frame; candidates are still obtained from

Table 2.6: Comparison of inter-person occlusion approaches

	hands			feet		
occlusion approach	P	R	F1	P	R	F1
individual	0.69	0.66	0.67	0.84	0.82	0.83
individual, pre-proc.	0.82	0.66	0.73	0.86	0.83	0.84
iterative, pre-proc.	0.87	0.62	0.72	0.91	0.78	0.84
joint, single-frame	0.90	0.64	0.75	0.97	0.85	0.91
joint, multi-frame	0.92	0.65	0.76	0.98	0.85	0.91

candidate tracklets obtained from multiple frames. Table 2.6 shows that as we increase the complexity of occlusion reasoning, performance increases. The only case in which F1 remains fixed or is lower is when comparing the individual and iterative occlusion handling approaches; in this case, obtaining joint solutions iteratively increases precision significantly, but also lowers recall. Our joint approach increases both precision and recall over the iterative approach, and obtains the highest F1 scores. A comparison between multi-frame and single-frame joint methods shows that our temporal transition factors improve results, but very little; this might be explained by the fact that some temporal information is included during low-level tracklet formation.

Chapter 3

Multi-agent event recognition in structured scenarios

3.1 Overview

The automated analysis of multi-agent activity is difficult due to interactions that lead to large state spaces and complicate the already uncertain low-level processing. Often, activities must satisfy rules that impose a spatio-temporal structure. This structure can be leveraged to disambiguate amongst complex activities. For example, in the case of one-on-one basketball, offensive and defensive rebounds are often ambiguous, since both players are near each other as they reach for the ball. However, the rules of half-court basketball can reduce this ambiguity by relating the rebound event to other less ambiguous events; e.g., if the ball were shot shortly after the rebound without any of the players running back to the three-point line, then an offensive rebound must have occurred, since a defensive rebound requires the ball to be *cleared* first (i.e., taken to the three-point line).

Our goal is to create a framework that, given a semantic description of what generally happens (i.e., rules, meaning of relevant events), determines the events that occurred. We test our framework on one-on-one basketball games, in which only two players interact, but event structure is non-trivial, and visual recognition is hampered by players frequently occluding each other. We do not use human annotations such as text, camera movement, shot-changes, or overlaid statistics (unlike

[21, 71]), which are typically used to analyze sports videos, as we seek a framework that can analyze broader classes of human/object interactions. The stationary camera simplifies image to court registration, but it also removes important information that a human operated camera provides; e.g, camera movement can reveal possession, and shot-changes provide a partial temporal segmentation.

We analyze single camera videos of one-on-one basketball in the context of court annotations (i.e., hoop and points on the court plane), and spatio-temporal relations describing the rules and events of interest. We automatically detect and track players, their hands and feet, and the ball, generating a set of trajectories which are used in conjunction with spatio-temporal relations to generate event observations. Knowledge about spatio-temporal event structure is expressed in first-order logic using a principled and extensible approach based on Allen's Interval Logic [1]. Robustness to low-level observation uncertainty is provided by Markov Logic Networks (MLN) [12], which attach weights to first-order logic formulas and dynamically construct Markov networks representing hypothesized events, spatio-temporal relationships, and low-level observations. Inference on this Markov network determines the high-level basketball events (e.g., *check*, *dribble series*, *shot*) that occurred.

Our main contribution is a system that efficiently and robustly recognizes events in structured scenarios from noisy visual observations by combining (1) visual analysis of people and object movements, (2) a powerful and natural event reasoning representation based on Allen's Interval Logic, (3) probabilistic logical inference via MLNs, and (4) efficient bottom-up event hypothesis generation.

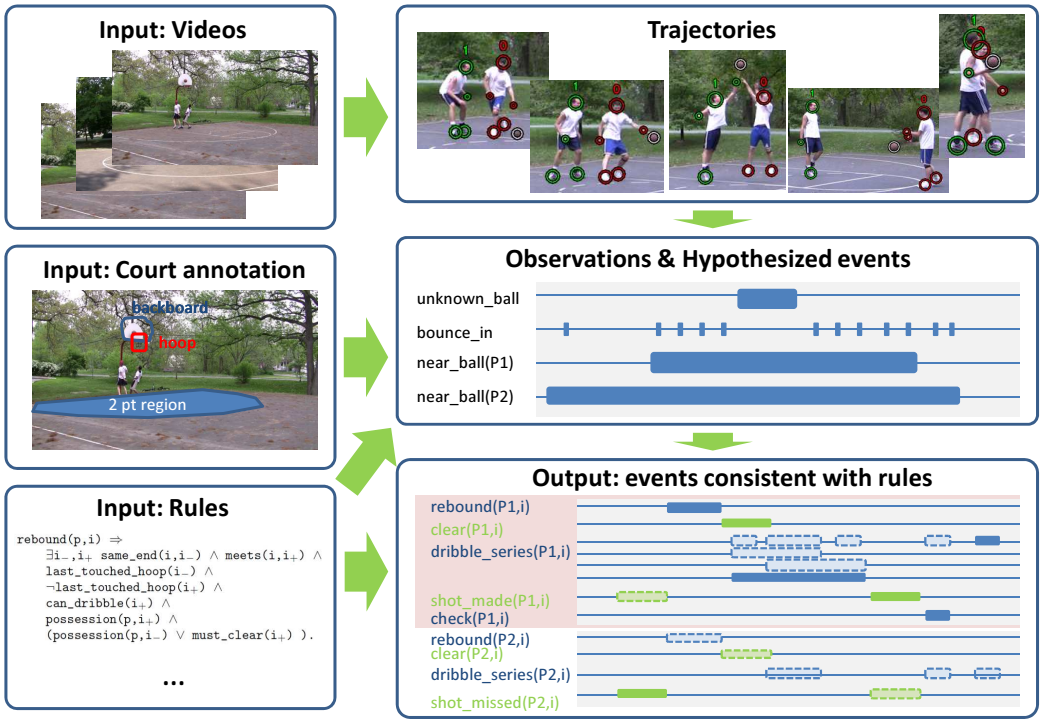


Figure 3.1: Framework overview.

3.2 Related Work

Hidden Markov Models (HMMs) [46] have been successfully applied to action recognition tasks, but their performance degrades as the state size increases (much more data is needed to train an accurate model); this is a problem, since multi-agent interaction models generally require a large state space. To deal with this complexity in highly coupled T'ai Chi hand movements, Brand *et al.* [6] presented coupled HMMs, which factorize the joint transition table into two smaller transition tables. Shi *et al.* [57] used Propagation Nets (P-Net), an extension of Dynamic Bayesian Networks (DBNs) that models duration and can represent complex activities including concurrent events, but requires manual specification of state connectivity.

Unfortunately, HMM and DBN extensions generally assume fixed number of actors and objects, do not handle missing observations well, and require large training sets to learn structure that a human could easily describe.

Expert domain knowledge has also been leveraged to create models of multi-agent activities. Intille and Bobick [26] recognize football plays by using temporal constraints to dynamically construct complex action Bayes nets from smaller manually specified Bayes nets that relate agent goals to visual evidence. Perse *et al.* [44] analyze team activities in basketball games by transforming trajectories into a sequence of semantically meaningful symbols and comparing them to sequence templates provided by domain experts. Ryoo and Aggarwal [53] model two person interactions by a context-free grammar (CFG), where high-level interactions are defined hierarchically using logical spatial and temporal predicates on sub-actions. Their atomic actions are detected using HMMs, but CFG parsing is not probabilistic and can be sensitive to low-level failures. Similarly, Store Totally/Partially Recognized Scenario (STRS/SPRS) [18, 64] approaches efficiently recognize multi-agent scenarios, but are symbolic and do not account for low level uncertainty. To introduce robustness to inconsistent first-order logic knowledge-bases (e.g., due to low-level errors, or imperfect rules), Tran and Davis [65] used Markov Logic Networks (MLN) [12] to analyze simple person-person and person-vehicle interactions. Similarly, Sadilek and Kautz [54] analyzed multi-agent interactions from GPS data, using MLNs to jointly denoise low-level data and incorporate temporally distant events. Their rules focus on a single event, *capture*, in the game of capture the flag.

Multi-agent activities have also been analyzed with little or no supervision.

Gupta *et al.* [21] use label data loosely associated with videos during training to automatically learn the spatio-temporal structure of baseball plays. Siracusa and Fisher [61] infer the interaction dependency structure in basketball games using a directed temporal interaction model and a latent variable to allow interaction dependency structures to change over time. While the latent variable state sequence is sampled by Markov Chain Monte Carlo (MCMC), given a state sequence, the posterior over dependency structures is obtained efficiently by exact inference. Sridhar *et al.* [62] perform unsupervised learning of events by modeling interactions between tracks as a relational graph structure that captures spatio-temporal relationships, clustering events by MCMC. These methods can be useful in learning multi-agent event patterns, but require large training sets to learn constraints.

Our approach leverages expert domain knowledge, expressed in first-order logic, and uses a powerful and natural representation based on Allen's Interval Logic to reason about complex relationships between multiple properties, events, and observations. By performing logical inference probabilistically using MLNs, our approach is robust to mistakes and knowledge base inconsistencies. A theoretical advantage of MLNs is that they can represent (and augment) popular formalisms such as BNs, HMMs, DBNs, and CFGs. Because of the expressiveness of our approach, we do not require large training sets; in fact, in our experiments, all knowledge is provided manually via rules, though probabilistic observations can be incorporated as in [65].

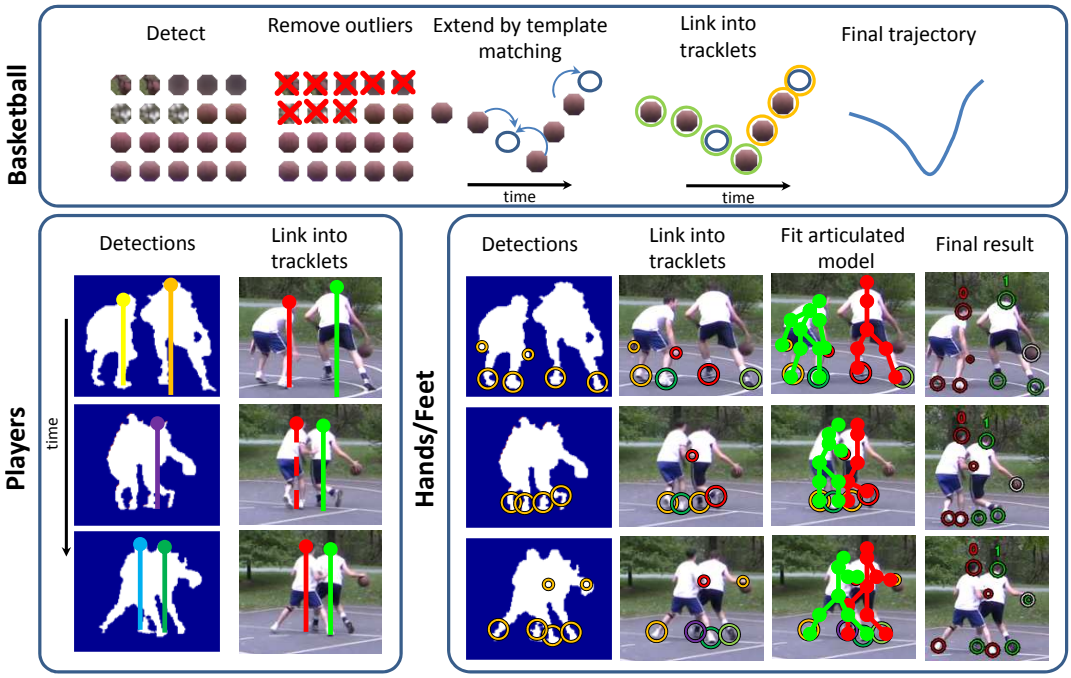


Figure 3.2: Trajectory extraction.

3.3 Trajectory Extraction

The low-level part of our system detects and tracks the basketball, players, and their hands and feet, providing their trajectories to the high-level component (see figure 3.2). Videos are preprocessed by first computing optical flow [42] and then detecting moving foreground pixels using background subtraction [30]. To handle slow outdoor lighting changes caused by clouds and changes in relative position of the sun, we split videos into smaller segments and use the same segment for training and testing, using flow to mask out moving pixels during the training phase. Once foreground pixels are obtained, we detect and track player heads, hands, and feet as in chapter 2, and deal with the ball similarly, first detecting candidates and then linking them into tracklets using a data association approach.

The ball is modeled as a circular object, so initial detections are found by fitting ellipses to foreground blobs and keeping only roughly circular blobs. As the ball is often in the air during a basketball game, this is a good way to learn its appearance. Detections, normalized to the same size, are clustered using k-center clustering [19] of color rank vectors (r', g', b') stacked to create a vector of size $3N$ for the N pixels belonging to the ball. Color rank provides some invariance to illumination changes and is computed by replacing each channel value with the percentage of pixels in the ball mask that have lower values for that channel. K-center clustering minimizes the maximum cluster radius, so outliers are placed in small clusters and can be removed. Remaining detections are extended forward and backward in time by template matching until the template no longer overlaps with the foreground mask or is sufficiently close to an existing detection. The resulting detections are linked into tracklets (with max frame gap of 1), keeping for each frame only the tracklet that contains the largest number of initial detections that were detected by ellipse fitting.

3.4 Event reasoning

Given the ball and player trajectories, court homography, and hoop location, we use the rules of basketball to generate and evaluate hypothesized events. The rules of the game are expressed using first order logic following the example of Allen *et al.* [1], and describe events by modeling their interactions with properties of the world (see table 3.1). Observations computed from the trajectory and court

annotations are incorporated into the knowledge base using a set of soft rules. To avoid computational complexity, instead of considering all $O(T^2)$ possible intervals for each event, our system uses the rules themselves to generate bottom-up event hypotheses from observations, aiming for a high recall ratio, while avoiding events that are unlikely given our rules (e.g., if the rules say that for a shot to occur, the ball must be in the air, then a hypothetical `shot` event is generated only when the ball is observed in the air). This bottom-up process may not always generate events that are consistent with the rules (the ball being in the air does not necessarily mean that a shot was attempted), so we use a probabilistic inference approach to determine which set of hypothesized event candidates most likely occurred, given the observations and the rules. See figure 3.1 for an illustration of this process.

3.4.1 Interval logic representation

The rules of basketball are non-trivial, even for the one-on-one case, so we need a principled approach to representing the rules and how they relate both to the state of the game and to visual observations. For this purpose, we adopt a framework similar to that proposed by Allen *et al.* [1], where predicates are grouped into three categories: properties, events, and actions. Temporal relationships between these predicates, which are defined on time intervals, are expressed using the following base binary relations and their inverses: `before`, `meets`, `overlaps`, `starts`, `during`, `finishes`, and `equals`. Properties describe the relevant parts of the state of the world; events change these properties when they occur, as long as prerequisite

properties hold before the event's occurrence; finally, actions are *programs* that an agent (such as a robot) executes in order to cause events to occur. This last category makes use of a *Try* predicate which indicates that an agent attempts to perform an action, which if successful, brings about one or more events. In our case, the system is a passive observer, so it cannot perform actions to bring about changes; thus, we ignore the *action* category described in [1]. Instead, we explicitly model observations with rules that generally hold true (but not always, due to mistakes in visual analysis, or because these rules are *rules-of-thumb*). Below we describe the categories of predicates and related axioms.

3.4.1.1 Properties

Properties describe the state of the world. In one-on-one basketball, the relevant properties are `possession`(p, i), `last_touched_hoop`(i), `can_dribble`(i), `must_clear`(i), and `must_check`(i) (see table 3.1 for descriptions).

3.4.1.2 Events

An event is defined by the prerequisite values of relevant properties prior to, during, and after its occurrence. The occurrence of an event could also imply that a related event occurred or that other events could not have occurred. Allen *et al.* [1] group event related axioms into *event definition*, *event generation*, *action definition*, and *event explanation closure* categories. We adopt these categories, excluding *action definition*, and add another category, *event mutual exclusion*. The *event definition*

Properties	
possession(p, i)	player p has possession during interval i; mutually exclusive and exhaustive between players
last_touched_hoop(i)	no player touched the ball since it last hit the hoop
can_dribble(i)	once dribble series ends, ball cannot be dribbled again until after a rebound, check, or steal
must_clear(i)	after a defensive rebound, ball handler must clear the ball by taking it to three-point line
must_check(i)	ball must be checked to player who has possession after a shot is made or ball goes out of bounds
Events	
shot_$\{*\}(p, i)$	the shot events, shot_made and shot_missed
dribble_series(p, i)	complete series of continuous ball bounces performed by player p during interval i
check(p, i)	sequence of passes to/from offensive player p, who is outside three-point line; last pass to p resumes play
rebound(p, i)	begins when ball falls from hoop (after a missed shot), and ends when player p obtains ball
clear(p, i)	after a defensive rebound player p clears ball by taking it to the three-point line
steal(p, i)	player p steals ball from other player, not by a rebound or out_of_bounds event
out_of_bounds(p, i)	starts when ball is out of bounds and ends when the ball is brought back on the court
Observations	
obs_in_air(i)	ball is in the air; implies that someone took a shot
obs_possession(p, i)	implies possession(p, i) with weight prop. to # frames ball is nearest p and p is farthest from hoop
obs_shot_made(p, i)	ball seen in air, obs_possession(p, i_-) is true before shot, obs_near_hoop(i_+) is true at end of shot
obs_shot_missed(p, i)	ball seen in air, obs_possession(p, i_-) is true before shot, obs_near_hoop(i_+) is not true at end of shot
obs_check(p, i)	sequence of passes with ball ending near p (pass observed by switches in obs_nearest_ball(p, i))
obs_dribble(p, i)	at least one bounce near p was observed

Table 3.1: Property and event predicates are used as queries. Observation predicates used as evidence in observation rules are shown; others such as obs_near_hoop, obs_near_ball, etc., are not listed.

axioms are of the form $\texttt{event}(i) \wedge \phi \Rightarrow \psi$, where ϕ and ψ are expressions that contain temporal constraints between the event and relevant properties. For the `rebound` event,

$$\begin{aligned}\texttt{rebound}(p, i) \Rightarrow \quad & \exists i_-, i_+ \texttt{same\_end}(i, i_-) \wedge \texttt{meets}(i, i_+) \wedge \\ & \texttt{last\_touched\_hoop}(i_-) \wedge \neg \texttt{last\_touched\_hoop}(i_+) \wedge \\ & \texttt{can\_dribble}(i_+) \wedge \texttt{possession}(p, i_+) \wedge \\ & (\texttt{possession}(p, i_-) \vee \texttt{must\_clear}(i_+))\end{aligned}$$

is an *event definition* axiom which states that for the `rebound` event to occur over interval i, person p first touches the ball at the end of the rebound event, and can then dribble the ball; p has possession after the rebound, and if p did not initially have possession, then the ball must be cleared. The *event generation* axioms are of the form $\texttt{event}(i) \wedge \phi \Rightarrow \exists i' \texttt{event}'(i') \wedge \psi$. In the basketball scenario, such a rule might say that if a shot event occurs, either a jump-shot, layup, or set-shot occurs. The *event explanation closure* axioms encode the assumption that only known events change properties, so if a property changed, an event affecting this property must have occurred. For example,

$$\begin{aligned}\texttt{can\_dribble}(i') \wedge \quad & \neg \texttt{can\_dribble}(i) \wedge \texttt{meets}(i', i) \Rightarrow \\ & \exists p, i'' \texttt{dribble\_series}(p, i'') \wedge \texttt{meets}(i'', i)\end{aligned}$$

states that for `can_dribble` to change from true to false, a dribble event must have occurred, after which the property transitions from true to false. Finally, *event mutual exclusion* axioms (not explicitly included in [1]) encode the constraint that some events cannot occur simultaneously.

$$\begin{aligned}\texttt{intersects}(i_1, i_2) \wedge \quad & \texttt{dribble\_series}(p_1, i_1) \wedge (i_1 \neq i_2 \vee p_1 \neq p_2) \Rightarrow \\ & \neg \texttt{dribble\_series}(p_2, i_2)\end{aligned}$$

This axiom states that a person can only take part in one `dribble_series` event at one time, and only one person at a time can dribble. The temporal relation

`intersects` is a disjunction of a subset of the base temporal constraints and their inverses that is true if the intersection of the intervals results in a time interval of some positive length.

3.4.1.3 Observations

Observations about the world could imply certain events happened, or that certain values of properties are more likely than others. These were not included explicitly in Allen *et al*. [1], but we include them since they determine the likelihoods of events that occurred.

$$\texttt{obs\_nearest\_ball}(p, i) \wedge \neg \texttt{obs\_nearest\_hoop}(p, i) \;\; \Rightarrow \;\; \texttt{possession}(p, i)$$

These rules may be inconsistent for two reasons: 1) observations are generated by video processing, which may include mistakes, and 2) some observation rules encode *common sense* knowledge that generally holds, but may at times lead to an inconsistent knowledge base. In the example above, when the ball is nearest the player who is farthest from the hoop, it generally means that player has possession of the ball, but this may not always be true (e.g., immediately after a rebound, or after a successful drive to the hoop). These potential inconsistencies are dealt with by our inference approach by allowing these rules to be treated as soft rules (i.e., a truth assignment can break these rules and incur a relatively low cost compared to, say, violating the rules of basketball).

3.4.2 Bottom-up event hypothesis generation

To avoid the computational cost of considering all $O(T^2)$ intervals for each event and property (assuming T frames), we generate a set of intervals which is small but has a high recall rate. Since the observation predicates are deterministic given the trajectories obtained by the tracking module, the intervals during which observation predicates hold are deterministic as well. Our approach, then, is to use the logic rules themselves to generate candidate intervals for events and properties. For example, since a `dribble_series` event is related by the observation rules to predicates such as `bounce_in`(i) and `near_ball`(p, i), we can use these predicates to generate hypothesized start and end times for a dribble series. Figure 3.3 depicts this process for the `dribble_series` event. Given the observation predicates, a small set of start times and end times is created, and from these two sets, a small set of intervals is created by pairing start and end times that are consistent with each other (end time is after start time, etc). A similar process is performed for all of the events shown in table 3.1: observed start and end times for observation predicates are used to create hypothesized start and end times for event predicates. We treat all ground atoms that contain the hypothesized intervals generated here as *open world* atoms; all others are *closed world*, and are assumed to be false.

The set of hypothesized intervals for each event predicate and the event explanation closure axioms presented earlier are then combined to generate times during which properties *might* change value. For example, if all properties change values either at the beginning or end of an event, then all unique hypothesized event start

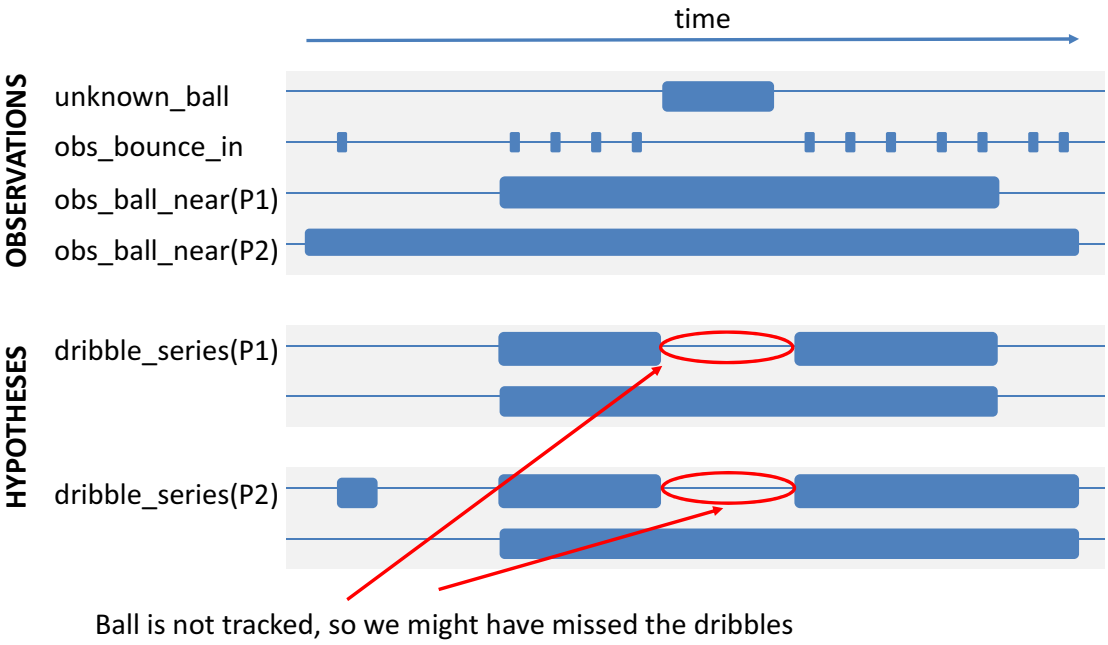


Figure 3.3: The bottom-up event generation module generates hypothesized event intervals from low-level observations, with the goal of achieving a high recall rate with a reasonably small set of intervals. Here, seven intervals are generated from observations for which $\texttt{dribble\_series}(p, i)$ is open world (could be true).

and end times can be collected to discretize time (non-uniformly). The intervals considered for properties are those intervals with start and end times that are consecutive in the ordered list of event interval start and end times (these time periods are called *moments*); thus, if there are M unique times that appear as start or end times in hypothesized event intervals, there will be $M + 1$ moments over which property predicates can be grounded. This allows us to remove some existential quantifiers (which can be computationally expensive). For example, the first line of the rebound event definition in section 3.4.1, can be replaced by the following:

$$\texttt{rebound}(p, i) \wedge \texttt{started\_by\_moment}(i, i_-) \wedge \texttt{meets\_moment}(i, i_+) \Rightarrow$$

	TP	FP	FN	P	R	F1
Tracking						
Hand tracklets	2153	1491	1214	.59	.63	.61
Foot tracklets	3821	1229	545	.76	.86	.81
Hand assign.	2095	475	1255	.82	.61	.70
Foot assign.	3600	339	720	.91	.82	.86
Ball tracks	1059	76	87	.93	.92	.93
Ball bounces	183	2	65	.99	.74	.85
Hypotheses						
Check	103	20	11	.84	.90	.87
Clear	122	282	15	.30	.89	.45
Dribble	226	436	27	.34	.89	.49
OutOfBounds	26	61	6	.30	.81	.44
Rebound	153	426	6	.26	.96	.41
ShotMade	75	559	2	.12	.97	.21
ShotMissed	166	468	5	.26	.97	.41
Steal	2	50	2	.04	.50	.04
Overall	**873**	**2302**	**74**	**0.27**	**0.92**	**0.42**

Table 3.2: Performance of tracking and event hypothesis generation.

3.4.3 Probabilistic Inference using Markov Logic Networks

The knowledge base is likely to contain inconsistencies, either due to noisy or missed observations, or due to imperfect rules that occasionally do not hold. For this reason, we relax these rules and perform queries probabilistically using Markov Logic Networks (MLN) [12]. Markov Logic Networks relax first-order logic by attaching a weight to each formula, such that when a world violates a formula, that world becomes less probable instead of becoming impossible. More formally, Domingos *et al.* [12] define an MLN as follows. An MLN consists of a set of first-order logic formulas F_i, associated real weights w_i and a finite set of constants $C = \{c_1, c_2, \ldots, c_{|C|}\}$. An MLN can then be viewed as a *template* for dynamically constructing a Markov network, given a set of constants. For a given set of constants, C, the network is constructed by creating one binary node for each grounding of each predicate,

which takes value 1 if that ground predicate is true and 0 if it is false. Each possible grounding of each formula F_i will then have an associated feature, which will have a value of 1 if that formula is satisfied, and 0 if it is not. Each feature will have an associated weight w_i. In the factor graph representation of the Markov network, ground predicates become nodes and formulas become factors defined over these nodes. The probability distribution of a world x is then given by $P(X = x) = \frac{1}{Z} \exp\left(\sum_{i=1}^{F} w_i n_i(x)\right)$, where $n_i(x)$ is the number of true groundings of formula F_i in x, F is the total number of formulas, and Z is a normalizing constant. Inference on this Markov network can then be performed using standard techniques. One theoretically desirable property of MLNs is that many common AI problems can be mapped to an MLN representation, including Bayes networks and Hidden Markov Models (HMM).

Predicates and formulas in our application contain three types of variables – moment, interval, and person – so the set of constants will include the $M+1$ unique moments, the I intervals, and the two players. Although weights can be learned for each formula, we manually set the weights using intuitive values; for example, formulas or axioms that describe constraints imposed by basketball rules have high weight, but *common sense* or observation formulas have lower weight, as there is a larger chance that they could cause the knowledge base to be inconsistent. We use Alchemy[1] [31] to generate ground MLNs, and AND/OR Branch-and-Bound [36] to perform exact inference.

[1]http://www.cs.washington.edu/ai/alchemy

3.5 Experiments

We demonstrate our approach on a dataset consisting of 7 outdoor sequences of one-on-one basketball (roughly 100,000 frames at 30fps, or 1hr of video), with varying camera positions, and 7 unique players. These videos contain varying illumination conditions (two are collected right before sunset and contain strong shadows), and 5 out of the 7 sequences contain full games to 11 points. The static annotation provided by the user includes hoop and backboard polygons and the homography from camera view to court plane (using 5-7 pairs of points on the court). Our framework does not yet include formulas to handle player identity switches, so additional human input is needed after player tracking to merge/split tracklets. Head location tracklets are shown to the user in an X-T plot, so that as many as 1600 frames can be inspected at a time. The user does not add or modify detections, but provides only identities of tracklets where necessary to prevent merges/splits.

For evaluation, ground truth is provided manually and includes locations of visible hands, feet, and ball every 100th frame, and start/end times of the events of interest. For hand, foot, and ball tracklets we first use the Hungarian algorithm to associate one ground truth location to one detected location, subject to some maximum distance, and then count the number of true/false positive and false negative matches. We use a threshold of $.1h$ for hands and feet (h is the height of the person), and $.5r$ for the ball (r is the ground truth radius of the ball). To evaluate events, we represent ground truth and detected events by intervals and use the Hungarian algorithm to match intervals to each other, minimizing the sum of

the absolute difference between their start/end times. Matches are disallowed if the gap between two intervals is too long (60 frames/2 sec).

Table 3.2 shows the overall performance of the tracking module, as well as the performance of event hypothesis generation. Tracking performance is good (feet and ball are the best, since hands are subject to large amounts of self-occlusion and fast motion). Event generation is evaluated by assuming all hypothesized event intervals are true; as expected, the recall ratio is high (.92), but some events are still missed, and a large number of false positives are present. High-level inference should be able to discard most false positives (increasing precision), but false negatives (missed intervals) are more problematic, since high-level inference only assigns truth values to hypothesized intervals; thus, final recall is strictly bounded by the recall of the event generation module. Table 3.3 shows the overall performance of our framework. As expected, most false positive hypotheses were removed (from 2302 to 279), but recall was reduced slightly since some true positive hypotheses became false negatives after being labeled *false* by the MLN inference, likely due to observation errors or missing nearby hypotheses that are required by the axioms. Table 3.4 shows the final event recognition performance, given tracking and hypothesis module performance, in order to analyze the sensitivity of our final result to varying input performance. Performance is relatively stable, except for sequences 1 and 7, which have much better ball tracks, and thus have the highest F1 scores for event recognition; this is not surprising since the ball is the most important object in the game.

Our formulas relate only events that are nearby in time, leaving long-term

	TP	FP	FN	P	R	F1
Check	102	19	12	.84	.89	.87
Clear	83	13	54	.86	.61	.71
Dribble	189	45	64	.81	.75	.78
OutOfBounds	21	3	11	.88	.66	.75
Rebound	115	71	44	.62	.72	.67
ShotMade	66	37	11	.64	.86	.73
ShotMissed	135	67	36	.67	.79	.72
Steal	2	24	2	.08	.50	.08
Overall	**713**	**279**	**234**	**.72**	**.75**	**.74**

Table 3.3: Overall event recognition performance

	Hands	Feet	Ball	Hyp.	MLN
Sequence 1	.68	.88	.96	.55	.99
Sequence 2	.50	.84	.79	.41	.60
Sequence 3	.59	.82	.79	.47	.76
Sequence 4	.70	.82	.74	.40	.74
Sequence 5	.76	.88	.80	.40	.67
Sequence 6	.76	.91	.81	.43	.69
Sequence 7	.76	.89	.88	.41	.89

Table 3.4: F1 scores of tracking, hypothesis generation (Hyp.), and MLN inference (MLN)

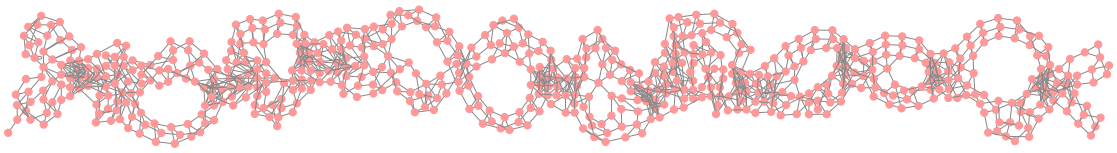


Figure 3.4: Ground MLN graph for 2910 frames, with 667 nodes, 2351 factors, and treewidth of 12. Nodes are ground predicates, and edges link nodes of ground predicates that appear in same formula.

relations to be implicitly enforced through properties, so the treewidth of the resulting ground MLN is relatively small (see Figure 3.4), enabling exact inference. The largest treewidth we encounter is 21, for a 25,287 frame sequence whose ground MLN contains 4,963 nodes and 18,440 factors, requiring 1.9 seconds for exact inference (not including tracking and network generation) on a 2.5 GHz Core 2 Quad CPU.

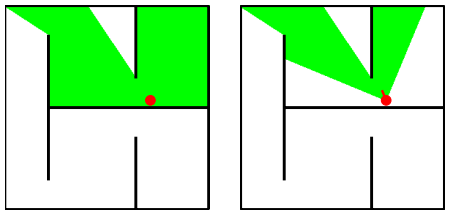
Chapter 4

Human activity understanding using visibility context

4.1 Overview

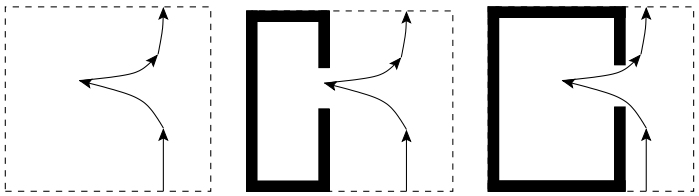
Visibility in architectural layouts affects human navigation, so a suitable representation of visibility context is useful in understanding human activity. Here, *visibility context* refers to a building's spatial layout visible to a human from various locations inside the building. See figure 4.1 for an illustration. Numerous studies in psychology and architecture have underscored the significant influence of a building's layout on the manner in which people navigate through it and emotively perceive it, e.g., [29, 16]. People walk through different parts of a building depending upon its layout and their purpose (e.g. to search, hide, explore).

The context provided by spatial layout may significantly affect an observer's interpretation of an agent's trajectory. Consider a scenario in which a person is



(a) (b)

Figure 4.1: Layout visibility with (a) omnidirectional and (b) directed view. The observer is denoted by the red circle, and the visible area – called an *isovist* – is the green polygon.



(a) (b) (c)

Figure 4.2: Part of a person's trajectory (foot-print) while searching for an object. Depending on the spatial layout, the sharp turn may be interpreted as a search point or the location at which the subject picked up the object.

navigating through a building. The person's objective is to find and pick up an object and then place it at some location. The observer is only provided with the person's trajectory (foot-prints) on the floor plan and is assigned the task of inferring the person's actions, such as whether the person was still searching for the object at a particular time, had already located it, etc. Figure 4.2 shows a zoomed in portion of a hypothetical trajectory generated in this scenario. Figure 4.2(a) shows the trajectory in the absence of any walls. In the absence of other information, the sharp turn in the trajectory could reasonably be interpreted as the location at which the person picked up the object – the person must have deviated from an otherwise straight path for a reason. Figure 4.2(b) shows the same trajectory, but with walls superimposed on the image. It is now much less likely that the agent picked up an object at that point; instead, it appears more likely that the person walked to that point only to explore the closed room and then moved on after discovering that the room did not contain the object. Now consider figure 4.2(c) – the same trajectory but with slightly different layout of the walls. In this case, the person walks deeper into the room. Now, it seems more likely that the person saw the object in the room

and walked in to pick it up – there would be no other reason to walk into an empty room. Thus, the same trajectory can be interpreted very differently based on the visibility context! This is the principal intuition of our work – how to represent a layout's visibility context and employ it for understanding human activity.

Consider a person searching for an object in a building. Two aspects of layout visibility influence the person's movement:

1. *Vantage points:* The person would give preference to locations that provide views of large parts of the building so that the search is efficient. The visibility context for the locations consists of features such as the field of view's area, perimeter, etc.

2. *Belief/memory:* While navigating through the building, the person builds a mental map of the areas already explored and those still to be investigated. A belief of the possible locations of the sought object is maintained and continuously updated with new information.

We present a Bayesian framework for jointly modeling the influence of visibility and belief on a person's movement. The person's goal, belief about the world, trajectory and visible layout are considered to be random variables that evolve with time during the movement. The belief/memory of the world and the visible layout constrain the person's goal. The belief and the goal together determine the sequence of actions taken by the person, which in turn determines the trajectory. Recognition is formulated as Maximum A Posteriori (MAP) estimation. The visibility context is represented with features based on behavioral studies of architecture. The features

are designed to enable generalization over novel spatial layouts.

Behavioral studies of architecture indicate that people's navigation through a building (spatial behavior) is closely coupled with the layout that is visible to them from different locations within the building. For example, Kaynar proposed that the spatial behavior of humans (in particular, their paths) in a museum can be predicted by visibility analysis [29]. The study indicated that presence or anticipation of unseen areas near a person's location is correlated with the change in the person's movement direction. Wiener and Franz showed that spatial behavior and experience can be predicted using measures derived from visibility context [16, 67]. For instance, measures of spatial qualities such as spaciousness, openness, complexity and order had significant correlations with the building's ratings given by human subjects.

Our proposed approach also relates to recent work in the robotics literature, where visibility – represented by isovists – has been used for motion planning in tasks such as exploration of unknown environments [5] and tracking a target in an environment with occlusions [4]. In the former example, a robot approximates its isovist using line-of-sight sensors and moves toward isovist boundaries that lead to unseen regions – the "inverse" of the problem we aim to solve. Rather than using visibility for motion planning, we instead use visibility to provide context in which an agent's motion can be interpreted.

Recent studies on human activity recognition have highlighted the importance of context provided by the scene, objects, etc. Our framework is closely inspired by the work of Baker et al. on the "inverse planning" problem of determining the intentions of an agent from trajectories [3]. They propose a Bayesian model for the agent's

intentions based on the trajectory and the spatial layout. The agent is assumed to always know the exact position of the object. They do not consider visibility – the spatial layout affects the analysis by constraining the possible movements. In our work, the person is searching for the object and therefore has to explore the layout. Moreover, only a part of the layout may be visible to the person at any given time. There are numerous computer vision studies in activity recognition that focus on trajectory-based features – we cite only a few, e.g., [50, 51]. These approaches do not model visibility context. In other studies, the scenes are manually pre-annotated to encode spatial and semantic information (e.g., doorways, hallways, furniture [56]). The proposed visibility context complements such approaches as it does not require explicit annotations, enabling generalization to novel scenes.

We illustrate the approach with experiments in a sprite-world domain. This isolates visibility and spatial layout as the only factors affecting an agent's actions. The trajectories are generated by a human performing search-based tasks in a virtual environment similar to first-person video game interfaces. We consider 6 layouts of varying complexity. To observe the generalization over layouts, the model is trained on 5 layouts and tested on the other, in a round robin format. As part of the experiments, human observers were asked to analyze the same trajectories with and without spatial information. Their scores are compared with that of the analysis performed by the Bayesian framework. The experiments show the importance of visibility context for activity recognition in our search-based task. Moreover, the proposed framework achieves average recognition rates comparable to those of the human observers.

This chapter is organized as follows. In section 4.2 we describe previous work on activity understanding and on effects that spatial and visibility constraints have on behavior. In section 4.4 we provide the visibility features that we use. In section 4.3 we discuss our model that incorporates visibility, memory, and belief. In section 4.5 we show the results of human experiments and of our proposed approach.

4.2 Related work

Baker et al. [3] propose a general Bayesian framework to explain how people reason and predict the actions of an intentional agent. They call their analysis of intentional reasoning "inverse planning" since they assume that agents build plans (sequences of actions) that achieve their goals, and to infer their intentions observers need only to invert a model of how goals affect plan formation. Using experimental results on pre-verbal infants from the cognitive science literature and their own experiments on humans, the authors motivate their Bayesian framework by noting that any model of intentional reasoning should include at least primitive planning capacities with the tendency to choose plans that achieve goals as efficiently as possible and that inferences about agents' goals should be probabilistic. In addition, motivated by how humans reason with the intentional stance, the authors introduce a utility function and assume that agents will prefer actions which lead to a larger expected increase in the utility function. In their design of the Bayesian framework, they place emphasis on the ability to learn from multiple environments and generalize to new ones.

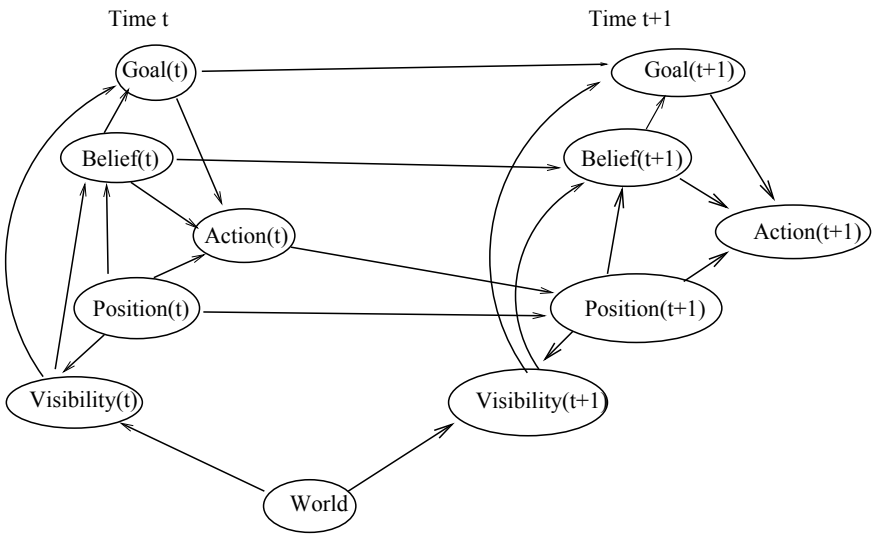


Figure 4.3: Graphical model for spatial behavior.

In their application of the general Bayesian framework to sprite-world inferences, Baker et al. introduce the assumptions that the world W is known to the agent and the observer (i.e. the agent knows the layout of the world and objects within it). However, agents often can only have partial observations of the world determined by what is visible from their current location, and at any point in time will know only the sections of the world that they have observed until that point. In this work, we will remove this assumption. Visibility constraints and memory are modeled in the Bayesian framework, eliminating the assumption that the agent has full knowledge of the world and incorporating spatial context into the model. Note that because spatial context is represented through visibility properties and is not represented directly by the environment, the model generalizes to new environments that have different spatial layouts.

4.3 Bayesian model for visibility in activity recognition

Consider an agent exploring and navigating in a world W. The agent's state at time t is defined to consist of three components:

1. Current goal, $g(t)$, for the movement. This controls the objectives guiding the agent's movements, e.g., searching for an object, approaching the object upon discovering it. An activity may in general consist of a sequence of goals, one leading to another.

2. The belief, $b(t)$, about the world. As the agent explores the world W, it continuously updates its belief about W based on the structure that is visible to it at any given time. The agent's belief consists of both memory as well as priors on the world's state.

3. The location of the agent in space, defined by $\mathbf{x}(t)$. The sequence of $\mathbf{x}(t)$'s forms the agent's trajectory. The location in space determines the substructure of the world that is visible to the agent. We denote the visible part of W by $v(t)$.

Based on its current goal, belief and position, an agent executes an action $a(t)$ with likelihood $p(a(t)|g(t), b(t), \mathbf{x}(t))$ to bring about a change in its location, $\mathbf{x}(t) \rightarrow \mathbf{x}(t+1)$. The action's outcome is modeled with the conditional probability $p(\mathbf{x}(t+1)|a(t), \mathbf{x}(t))$. The change in location provides a novel view of the world $v(t+1)$ subject to the likelihood $p(v(t+1)|\mathbf{x}(t+1), W)$. This in turn results in an updated belief, $b(t+1)$ with probability $p(b(t+1)|v(t+1), b(t))$. The belief and current world view may lead to a

change in the agent's goal, $g(t+1)$ with likelihood $p(g(t+1)|b(t+1), v(t+1), g(t))$. E.g., when the agent locates the sought object, the goal shifts from searching to that of approaching the object. The conditional probability structure is summarized in the graphical model in figure 4.3. In practice, we only observe the agent's trajectory, $\mathbf{x}(t)$. All other components of the agent's state are hidden variables.

The layout's visibility is represented by the visibility-polygon, also called an *isovist*. The visibility-polygon is defined by the walls of the scene that are visible to the agent from a particular location in the world, and the occluded edges. Isovist qualities, such as area, perimeter and presence of occluded edges, determine whether a location is a good vantage point for searching (these qualities will be discussed briefly in section 4.4). Suppose, the agent's goal is to search for an object, and it anticipates some location to give a good view of a large part of the world, then the agent would likely navigate towards that point. Therefore, if an agent is observed to show preference to locations with high visibility, then it is assigned a high likelihood to a search goal. On the other hand, if an agent is observed to walk a direct path to a corner in a room then it has most likely located the sought object and is proceeding to pick it up.

4.4 Modeling visibility

Motivated by observations from cognitive science on visibility and architecture as discussed above, we represent visibility, $v(t)$, by using isovists and features derived from them. Figure 4.1(a) shows an isovist which we refer to as a *full isovist* since

it includes all visible areas if viewing angle and field of view constraints are not taken into account. However, humans have a limited field of view; a *partial isovist* refers to an isovist that excludes all areas that are not in an oriented observer's field of view. Figure 4.1(b) shows a sample partial isovist. The edges of the isovist that do not coincide with a wall are referred to as *occluded edges*; they are formed when walls occlude an observer's view, and are often potential directions for further exploration. The isovist can be used directly for modeling what an observer sees along the trajectory, which facilitates the process of reasoning about the observer's belief of the world. However, features derived from the isovist contain additional information related to spatial layout that can further help observers reason about human behavior.

A variety of features can be computed from isovists (full and partial), many of which are discussed in [9, 16, 67]. In addition, visibility graphs (which can be used to compute shortest paths) are closely related to isovists, since each node in the visibility graph corresponds to a point in a scene and there is an edge in the graph between two nodes if they are visible (i.e. one node lies inside the other node's isovist). We calculate features using both isovists and visibility graphs.

The first group of features are derived from the isovist at the current location along the trajectory. They include isovist *area*, *perimeter*, *occlusivity* (sum of lengths of all isovist occluded edges divided by perimeter), *openness* (ratio of length of occluded edges to that of non-occluded edges), *compactness* (square of isovist perimeter divided by area), *minimum distance to an occluded edge* and *minimum distance to a wall*. The first five correspond to the spatial qualities of spaciousness,

openness, and complexity. The last two contain information about the agent's current positioning relative to the layout. Consider *area* for an example of how these features can be helpful: as the agent moves from one room to another, isovist area peaks when the agent is in the doorway between the two rooms, which can be helpful if certain events are more or less likely to occur in doorways.

The second group of features uses isovists and limited path history (such as positions at time t and $t - \Delta$), which show how agents have changed their visibility fields over time. The most straightforward use of limited path history is to approximate derivatives of an isovist field along the path. However, there are useful features that are not simply approximated derivatives. Such features include *new view area*, *lost view area*, and *deviation from the shortest path*. The new view area is the area of the isovist region at time t that does not coincide with the intersection of the isovists at times t and $t - \Delta$ (lost view area is computed similarly). Deviation from the shortest path uses visibility graphs instead of isovists, and is the additional cost relative to the shortest path that an agent must incur to travel from a start to an end point through a middle point. If the difference is large, then the middle point is a significant detour from the shortest path, and the agent likely incurred the additional cost because there was some reward for deviating from the shortest path.

Features in the third group are based on the complete history of the trajectory (e.g. the union of all areas seen by time t), and include *area seen ratio* (total area seen divided by total layout area), and *geodesic distance to unseen regions*. *Geodesic* distance is the shortest distance after taking walls into account. This is useful

because a rational agent who is exploring a region or searching for an object will tend to move toward some unexplored portion of the environment, thus causing the shortest distance to unseen regions to decrease.

Figure 4.4 shows how some features described above change as an agent performs a search-based task. The task in this case is to first search for and "pick up" a blue cube and then to search for and "pick up" a red cube, and is described in more detail in section 4.5.

4.5 Recognition

There are several human behaviors that are significantly influenced by the structure of scene layout, e.g., searching, hiding, stalking. We use searching activities as the domain for demonstrating the importance of visibility context and the proposed Bayesian framework. The trajectories were collected by a human agent navigating in a virtual 3D environment. The interface is similar to that commonly used in first-person video games. A virtual environment allowed accurate and precise observations of ground truth, and isolated the spatial visibility features to be the only factors influencing the movement. Six scenes were constructed, shown in figure 4.5. They have distinctive spatial structure, varying from cubicles as seen in offices to aisles commonly occurring in superstores.

There are a number of possible tasks that can be assigned to the human agent to investigate search activities. These can range in complexity from a very simple task, e.g., "Search for and pinpoint a stationary object", to relatively complex tasks

such as "Search for an object that is trying to evade you". There can also be variations such as the degree of background clutter, a sequence of search-and-locate subtasks, etc. We chose a search task of medium complexity:

1. At the start, the human agent is "teleported" to a random location in a scene. The task is to search out a blue cube placed randomly in scene, and go touch it. Then the agent must proceed to search for a red cube, also placed randomly, and touch it. There were no other objects except the blue and red cubes in the scene.

2. The recognition task for the observer is to estimate the location of the blue cube using just the trajectory of the agent. The recognition task was posed to human subjects as well as to the proposed Bayesian framework.

The reasoning is that:

- As the blue cube is placed completely at random, the only distinguishing feature for its location would be the trajectory of the agent before and after touching it.

- As the agent is tasked to search for the red cube after touching the blue one, the observer is forced to distinguish between searching and non-searching behavior. This is a harder recognition task than the case in which the agent is instructed to either walk to a predefined place or move around randomly. In the latter case, the purposive search for the blue cube would be easily distinguishable.

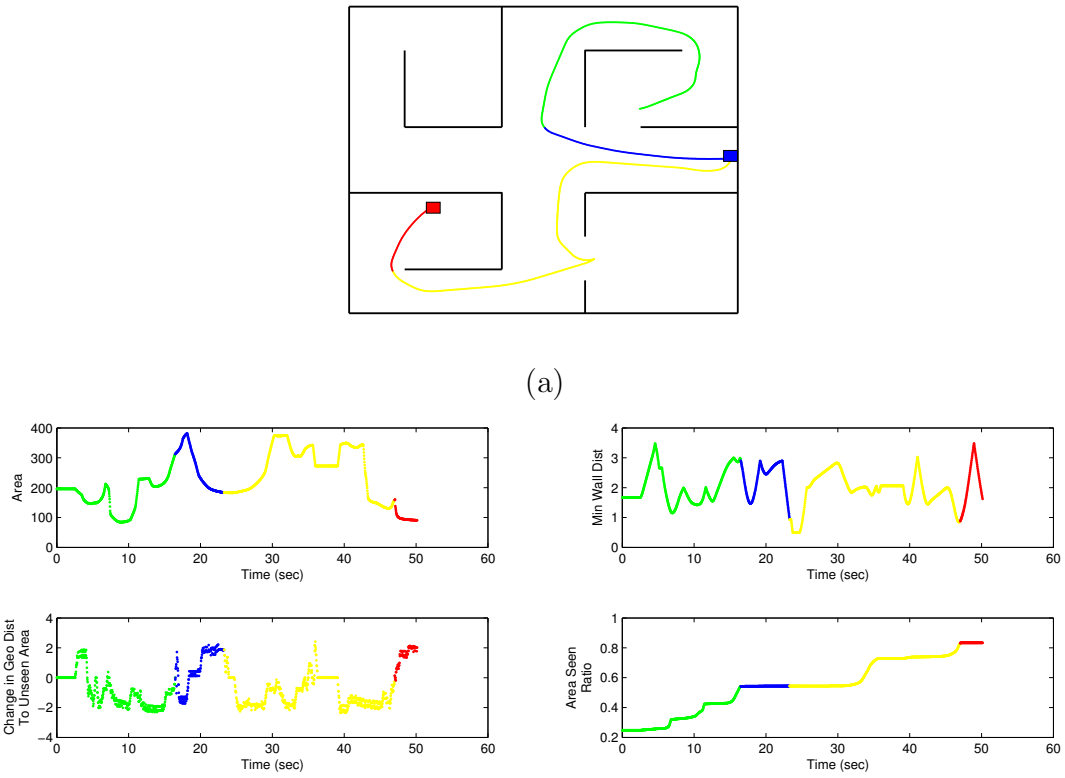


Figure 4.4: Sample trajectory and corresponding visibility features. In part (a), trajectory is shown segmented by search sub-goals: green = "search for blue cube", blue = "pick up blue cube", yellow = "search for red cube", red = "pick up red cube". Part (b) shows visibility features during the trajectory as they vary with time. The coloring corresponds to trajectory coloring. Note that when the blue cube is reached, *isovist area* and *minimum distance to a wall* are close to a local minimum, the *change in geodesic distance to unseen area* is positive, and the *seen area ratio* is relatively flat.

For each of the six scenes, the human agent was asked to perform the search task 15 times, generating 90 trajectories in total. Each time, the blue and red cubes were placed randomly.

To isolate and highlight the importance of visibility context, half the human subjects were asked to perform the recognition without any information about the walls present in the scene, and other half were shown the trajectories with the

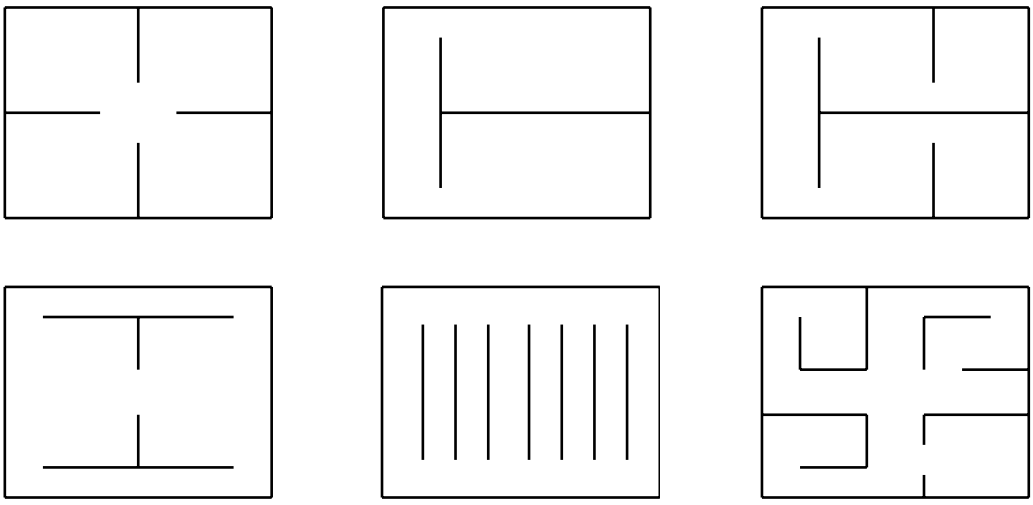


Figure 4.5: Scenes. The top and bottom rows show scenes 1 to 3 and 4 to 6, respectively.

walls correctly superimposed. The results indicate that human recognition performance improves substantially when the context of the surrounding is provided. The Bayesian framework was assigned the recognition task in the presence of visibility context. The results indicate that the approach's performance is comparable to that of humans.

4.5.1 Human recognition results

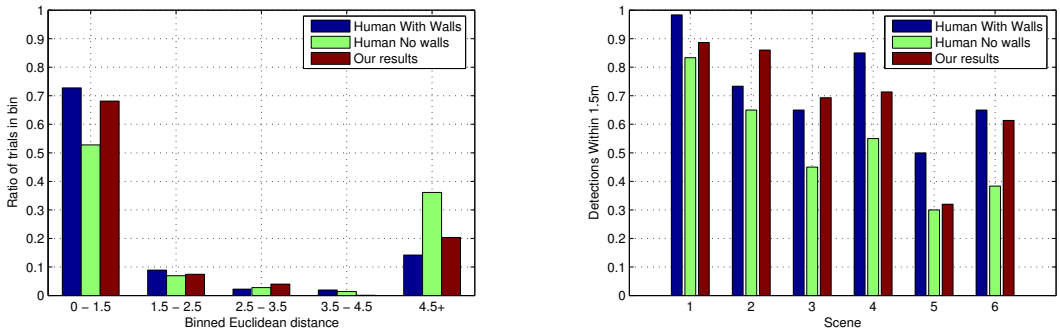
For the human subject experiments, 8 subjects were presented with all 90 trajectories. The subjects were split into two groups: one group was shown the walls and trajectory (the 'walls' group) and the other was shown only the trajectory (the 'no walls' group). The subjects were informed of the agent's task and were instructed to pick the location on the trajectory where the agent most likely picked up the blue cube. As figure 4.6 shows, the 'walls' group performed best, detecting 72.8%

of the blue cube pick up events within 1.5 meters of the ground truth (scenes are either 28m by 28m or 40m by 28m; see figure 4.7 for a depiction of error relative to scene size). The 'no walls' group detected only 52.8% within the 1.5m error margin. Thus, the visibility and spatial context of the walls provides significant information to humans for inferring the intention of the agent.

The recognition performance showed significant variation w.r.t. the scenes - see figure 4.6(b). Scenes 1, 2 and 3 have lower complexity of wall layout compared to Scenes 4, 5 and 6. Scenes 5 and 6 are especially difficult. The complexity in scene 5 arises from the fact that the room is divided into aisles, allowing the agent to walk directly through an aisle without returning after picking up an object in the aisle. This greatly decreases the performance of the humans with and without walls. The complexity of scene 6, however, arises from the number of small rooms that must be explored. Without layout information, subjects have no clue as to what caused all the turns in the trajectory. However, giving layout information to the subjects resulted in a much larger improvement in detection error for scene 6 compared to scene 5.

4.5.2 Recognition with the Bayes framework

The recognition task is formulated as MAP estimation of the location of the blue cube given the human agent's trajectory and the scene. We compute a set of intermediate goal-points based on high-curvature locations in the person's trajectory. Each of these goal-points is considered to be a hypothesized location for the blue cube.



(a) (b)

Figure 4.6: Experimental results: (a) Histogram of error between experimental results and ground truth, showing the proportion of detections that lie in each error range (in meters), and (b) ratio of detections with less than 1.5m error, grouped by scene.

The recognition result is the hypothesis with the highest likelihood, estimated using MAP. The joint likelihood of the agent's state, the sequence of executed actions and the world is

$$\begin{aligned} p(\mathbf{x}, b, g, a, W) \quad = \quad & \prod_{t=2}^{T} p(\mathbf{x}(t)|\mathbf{x}(t-1), a(t-1)) \\ & \prod_{t=1}^{T} p(v(t)|\mathbf{x}(t), W) \\ & \prod_{t=2}^{T} p(b(t)|v(t), b(t-1)) \\ & \prod_{t=2}^{T} p(g(t)|v(t), b(t), g(t-1)) \\ & \prod_{t=1}^{T-1} p(a(t)|\mathbf{x}(t), b(t), g(t)) \end{aligned} \tag{4.1}$$

The first 3 product-terms in the joint likelihood eq.(4.1) are determined from the trajectory, the scene's layout and the blue cubes hypothesized location. Thus, $p(\mathbf{x}, b, g, a, W)$, the confidence for the blue cube's location hypothesis is determined by $p(g(t)|v(t), b(t), g(t-1))$ and $p(a(t)|\mathbf{x}(t), b(t), g(t)) \propto \frac{p(g(t)|x(t),a(t),b(t))}{p(g(t)|x(t),b(t))}$ – the good-

ness of the goal sequence given the visibility and trajectory.

The scene and the blue cube's hypothesized location together determine the world W's state. Given the trajectory and W, the sequence of visible worlds $v(t)$ is computed. This, in turn, generates the sequences of beliefs, $b(t)$, of the human agent during the navigation (of course, conditioned on the hypothesized location). The beliefs and visible world together determine the sequence of goal states, $g(t)$, of the human agent:

- Until the time the person sights the blue cube, the goals, $g(t)$'s, can either be "search" - giving preference to high visibility areas, or "via-point" - that are just intermediate points to reach some other goal, e.g., to turn a corner.

- After sighting the blue cube and until the hypothesized time of touching the blue ball, the goal points must be "via-points". There is no searching required.

- After the hypothesized touching of the blue cube and until the sighting of the red cube, the goal points will either be "search" or "via-points".

- After sighting the red cube and until touching it (the end of the sequence), the goal points must be "via-points" because there is need for further search.

The likelihood for a goal-point, $g(t)$, to be a "search" is determined from the visibility field. Specifically, it is determined by the newly seen area. The likelihood for a goal-point, $g(t)$, to be a "via-point" is determined by the *deviation from shortest path*. Combining the log-likelihoods of these goal-points gives the likelihood of the sequence of goals before and after the hypothesized pickup of blue cube, denoted

by l. This must be combined with the likelihood of the goal-point at the blue cube location. A boosting algorithm is employed to classify correctly hypothesized blue cube locations from incorrect ones based on l and $v(t_b)$, the visibility fields at the blue cube location. During training, the negative class contains sequences of goals generated by blue cube locations that are known to be incorrect, and the positive class contains the sequences that are known to be correct. Thus, the classifier is trained to recognize the MAP hypothesis using the visibility features at the blue cube location and the likelihood of the goals at all other times. During testing, the most likely hypothesis is defined to be the one with maximum confidence.

Note that we avoid the complex task of Bayesian inference on the proposed network by brute search through the hypothesis space. The number of hypothesized points (less than 50 in our experiments) was significantly smaller than the total number of points in the trajectory.

We tested the proposed approach on the same 90 trajectories described above. Unlike the human subjects (who did not require training scenes!), our algorithm was trained on 5 scenes, and tested on the remaining scene following a round robin protocol. As in the human experiments, the algorithm computed the likely location of the blue cube in each scene. Since the algorithm is blind to the test scene, the experiments test the generalization of the framework to novel scenes. Figure 4.6(a) shows that the results of our algorithm are very good (68.1% of detections are within 1.5m of the ground truth). Figure 4.6(b) shows that our algorithm performed well for all scenes (always better than the 'no walls' group), and better than the average of the 'walls' human subjects group for scenes 2 and 3!

Figure 4.7 shows example detections by our algorithm (red circle), 'no walls' human group (blue cross), and 'walls' group (green x). In the first row, the detections by our algorithm and by the 'walls' human are correct, but the 'no walls' human chose the wrong sharp point in the trajectory since the subjects were not provided with spatial information. In the absence of spatial information, the locations chosen by the 'no walls' humans are reasonable choices. The second row shows cases where all three groups were able to locate the blue cube location. Finally, the third row shows examples of where our algorithm failed in locating the blue cube. In the two leftmost cases, the scene is difficult to interpret because of the aisles. In the rightmost image of the third row, the point that our algorithm chose could be mistaken for the blue cube position since agent did not immediately leave the room after reaching the entrance, but instead entered the room slightly before turning around.

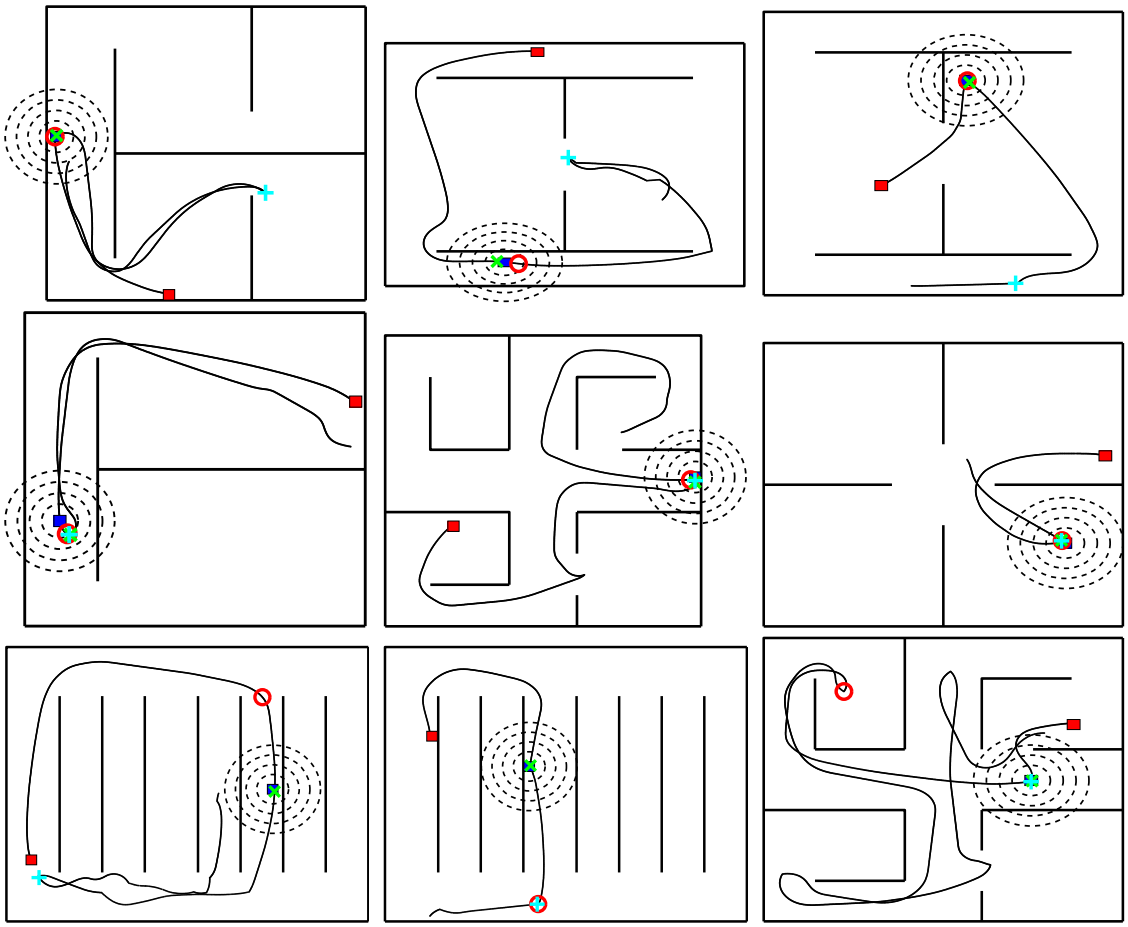


Figure 4.7: Sample trajectories with ground truth, human, and algorithm detections (red circle - our algorithm, cyan plus - human no walls, green x - human with walls, blue square - blue cube ground truth). Dotted circles denote errors of 1.5m, 2.5m, 3.5m, and 4.5m from ground truth. The top row shows cases in which humans with no walls chose the wrong solutions, while our algorithm and the humans with walls were able to select the correct blue cube location. The middle row shows cases where the locations were correctly chosen by both groups of humans and our algorithm. Finally, the third row shows cases in which our algorithm failed to locate the blue cube.

Chapter 5

Conclusion

This dissertation explored the problem of activity understanding based on trajectories of people and their extremities, providing contributions to the following three tasks: tracking the limbs of multiple interacting people (chapter 2), modeling the spatio-temporal structure of events (chapter 3), and modeling relationships between people and their environment (chapter 4). The combined contribution of this work to the field is the extension and strengthening of previous approaches by improved models of relationships and interactions between people, objects, and the scene. In real world scenarios, complex interaction models can lead to solutions that are expensive to compute and that are brittle in the presence of visual uncertainty. As a result, the contributions of this work include efficient computational models that incorporate probabilistic reasoning. The result is the increased ability of an automated computer vision system to function in challenging real world applications.

In chapter 2 we proposed a framework for detecting and tracking extremities of multiple interacting people. We quantitatively evaluated our approach on the publicly available HumanEva I dataset, a dataset of a group of interacting people,

and a dataset of one-on-one basketball games. Our experiments show that AND/OR Branch-and-Bound with lazy evaluation can significantly reduce computational cost, while yielding the globally optimal solution in each frame. Our approach is flexible enough to deal with significant occlusion of people in groups as well as rapid motions and large pose variations observed during basketball games. In chapter 3, we presented a framework which, given a semantic description of what generally happens in a scenario, uses video analysis and mixed probabilistic and logical inference to annotate the events that occurred. We demonstrated our approach on one-on-one basketball videos, recognizing complex events without additional cues such as text, camera movement, shot-changes, or overlaid time or score statistics. Because of the flexibility of the logical knowledge representation and relatively few restrictions on problem type (concurrent events are allowed, number of actors can vary, etc.), we believe that our framework can be extended to more difficult scenarios or other problem domains. Finally, in chapter 4, we presented a framework in which visibility context is utilized to aid in reasoning about human activity. Our experiments showed that features used to represent visibility generalize well over new scenes, and that our method resulted in a detection rate close to that of human observers in a search-based task.

Future directions for the work described in this dissertation include:

- The implementation of the tracking framework described in chapter 2 makes use of background subtraction for detecting and tracking people and their limbs. This work can be extended by incorporating stronger body part likelihoods that do not depend on segmentation. Alternatively, segmentation might be performed using other approaches that do not require a model of the background. Similarly, our deterministic occlusion constraints can be relaxed by incorporating a probabilistic occlusion model as in [13].

- Our event hypothesis generation approach is based on feed-forward processing of trajectories. If an event is never hypothesized, then MLN inference cannot infer that event. Can the space of possible event intervals be sampled in a better way without significantly affecting computational complexity?

- As described in chapter 3, a ground MLN is constructed for all observed data in a sequence. In an on-line streaming setting, memory and computational constraints require that only a subset of observations be used. How many observations are needed in such a setting? If too many are used, memory and inference cost will be high; if too few are used, results will be inaccurate.

- As the number of people increases, the assignment of roles (Player 1, Player 2, etc.) to tracks becomes increasingly challenging, especially in the presence of tracking errors, and must be dealt with carefully.

- In sufficiently complex scenarios, experts cannot manually specify all axioms/formulas. Thus, an interesting future direction is the use of approaches such as Probabilistic Inductive Logic Programming (PILP) [47] to automatically learn additional formulas given a background theory.

- Visibility context, described in chapter 4, is currently applied to trajectories created by humans performing a task in a virtual environment. In future work, it can be combined with the frameworks described in chapters 2 and 3 for the analysis of real world activities.

In summary, this dissertation presented contributions to human activity understanding based on trajectories of people and their hands and feet. By dealing with semantically meaningful trajectories, as opposed to relying only on low-level image features computed globally for an image or for the image regions around people, the proposed framework can naturally represent and reason about actions and interactions in terms of spatio-temporal relationships between body parts, objects, and the scene. In addition, the framework performs probabilistic reasoning at various stages of processing for robustness against noisy visual observations. Promising experimental results suggest that the proposed contributions enable efficient and accurate activity analysis which should scale well to more complex scenarios.

Bibliography

[1] James F. Allen and George Ferguson. Actions and Events in Interval Temporal Logic. *Journal of Logic and Computation*, 1994.

[2] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. *CVPR*, 2008.

[3] Chris Baker, Joshua B. Tenenbaum, and Rebecca Saxe. Bayesian models of human action understanding. In *NIPS*, 2005.

[4] T. Bandyopadhyay, Yuanping Li, M.H. Ang, and David Hsu. A greedy strategy for tracking a locally predictable target among obstacles. In *ICRA*, pages 2342–2347, 2006.

[5] Tirthankar Bandyopadhyay, Zheng Liu, Marcelo H. Ang, and Windston Khoon Guan Seah. Visibility-based exploration in unknown environment containing structured obstacles. In *ICAR*, pages 484–491, 2005.

[6] M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. In *CVPR*, 1997.

[7] William S. Cleveland and Susan J. Devlin. Locally weighted regression: An approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83(403):596–610, Sept 1988.

[8] Dorin Comaniciu and Peter Meer. Mean shift analysis and applications. In *ICCV*, 1999.

[9] Larry S. Davis and Michael L. Benedikt. Computational models of space: Isovists and isovist fields. *Computer Graphics and Image Processing*, 11(1):49–72, 1979.

[10] Rina Dechter. Bucket elimination: A unifying framework for probabilistic inference. In *UAI*, 1996.

[11] Rina Dechter and Robert Mateescu. AND/OR search spaces for graphical models. *Artif. Intell.*, 2007.

[12] Pedro Domingos, Stanley Kok, Hoifung Poon, Matthew Richardson, and Parag Singla. Unifying Logical and Statistical AI. In *AAAI*, 2006.

[13] M. Eichner and V. Ferrari. We are family: Joint pose estimation of multiple persons. In *ECCV*, 2010.

[14] Mark Everingham, Josef Sivic, and Andrew Zisserman. "Hello! My name is... Buffy" - automatic naming of characters in TV video. In *BMVC*, pages 889–908, 2006.

[15] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 2005.

[16] G. Franz and J. M. Wiener. Exploring isovist-based correlates of spatial behavior and experience. In *Proceeding of the 5th Space Syntax Symposium*, pages 503–517, 2005.

[17] Stephan Gammeter, Andreas Ess, Tobias Jäggli, Konrad Schindler, Bastian Leibe, and Luc Van Gool. Articulated multi-body tracking under egomotion. In *ECCV*, 2008.

[18] Malik Ghallab. On Chronicles: Representation, On-line Recognition and Learning. In *KR*, 1996.

[19] Teofilo F. Gonzalez. Clustering to minimize the maximum inter–cluster distance. In *Journal of Theoretical Computer Science*, 1985.

[20] Abhinav Gupta, Anurag Mittal, and Larry S. Davis. Constraint integration for efficient multiview pose estimation with self-occlusions. *PAMI*, 2008.

[21] Abhinav Gupta, Praveen Srinivasan, Jianbo Shi, and Larry S. Davis. Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *CVPR*, 2009.

[22] I. Haritaoglu, D. Harwood, and L.S. Davis. W4: real-time surveillance of people and their activities. *PAMI*, 2000.

[23] N. Howe. Evaluating lookup-based monocular human pose tracking on the HumanEva test data. In *EHuM*, 2006.

[24] Gang Hua, Ming-Hsuan Yang, and Ying Wu. Learning to estimate human pose with data driven belief propagation. In *CVPR*, 2005.

[25] Chang Huang, Bo Wu, and Ramakant Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV*, 2008.

[26] Stephen S. Intille and Aaron F. Bobick. A framework for recognizing multi-agent action from visual evidence. In *AAAI*, 1999.

[27] Hao Jiang and D.R. Martin. Global pose estimation using non-tree models. In *CVPR*, 2008.

[28] Kalev Kask and Rina Dechter. Mini-bucket heuristics for improved search. In *UAI*, 1999.

[29] Ipek Kaynar. Visibility, movement paths and preferences in open plan museums: An observational and descriptive study of the ann arbor hands-on museum. In *Proceeding of the 5th Space Syntax Symposium*, 2005.

[30] Kyungnam Kim, Thanarat H. Chalidabhongse, David Harwood, and Larry S. Davis. Real-time foreground-background segmentation using codebook model. *Real-Time Imaging*, 2005.

[31] Stanley Kok, Parag Singla, Matthew Richardson, and Pedro Domingos. The Alchemy System for Statistical Relational AI. Technical report, Dept. of Computer Science and Engineering, University of Washington, Seattle, WA, 2005.

[32] Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008.

[33] C.-S. Lee and A. Elgammal. Body pose tracking from uncalibrated camera using supervised manifold learning. In *EHuM*, 2006.

[34] Victor Lempitsky, Andrew Blake, and Carsten Rother. Image segmentation by branch-and-mincut. In *ECCV*, 2008.

[35] Yuan Li, Chang Huang, and Ram Nevatia. Learning to associate: Hybrid-Boosted multi-target tracker for crowded scene. In *CVPR*, pages 2953–2960, 2009.

[36] Radu Marinescu and Rina Dechter. AND/OR Branch-and-Bound search for combinatorial optimization in graphical models. *Artif. Intell.*, 2009.

[37] J. Martinez del Rincon, J.C. Nebel, D. Makris, and C. Orrite. Tracking human body parts using particle filters constrained by human biomechanics. In *BMVC*, 2008.

[38] Robert Mateescu and Rina Dechter. Mixed deterministic and probabilistic networks. *ICS Technical Report (Submitted)*, 2008.

[39] Joris M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, August 2010.

[40] Vlad I. Morariu, V. Shiv Naga Prasad, and Larry S. Davis. Human activity understanding using visibility context. In *IEEE/RSJ IROS Workshop: From sensors to human spatial concepts (FS2HSC)*, 2007.

[41] Vlad I. Morariu, Balaji Vasan Srinivasan, Vikas C. Raykar, Ramani Duraiswami, and Larry S. Davis. Automatic online tuning for fast Gaussian summation. In *NIPS*, 2008.

[42] Abhijit S. Ogale and Yiannis Aloimonos. A roadmap to the integration of early visual modules. *IJCV*, 2007.

[43] Sangho Park and J. K. Aggarwal. Simultaneous tracking of multiple body parts of interacting persons. *CVIU*, 2006.

[44] Matej Perse, Matej Kristan, Stanislav Kovacic, Goran Vuckovic, and Janez Pers. A trajectory-based analysis of coordinated team activity in a basketball game. *CVIU*, 2009.

[45] R. Poppe. Evaluating example-based pose estimation: Experiments on the HumanEva sets. In *EHuM2*, 2007.

[46] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Readings in speech recognition*, 1990.

[47] Luc De Raedt and Kristian Kersting. Probabilistic Inductive Logic Programming. In *ALT*, 2004.

[48] D. Ramanan, D.A. Forsyth, and A. Zisserman. Tracking people by learning their appearance. *PAMI*, 2007.

[49] Xiaofeng Ren, A.C. Berg, and J. Malik. Recovering human body configurations using pairwise constraints between parts. In *ICCV*, 2005.

[50] P. Ribeiro and J. Santos-Victor. Human activities recognition from video: modeling, feature selection and classification architecture. In *HAREM 2005 - in conjunction with BMVC 2005*, pages 61–70, 2005.

[51] Neil Robertson and Ian Reid. A general method for human activity recognition in video. *Comput. Vis. Image Underst.*, 104(2):232–248, 2006.

[52] Remi Ronfard, Cordelia Schmid, and Bill Triggs. Learning to parse pictures of people. In *ECCV*, 2002.

[53] M. S. Ryoo and J. K. Aggarwal. Recognition of Composite Human Activities through Context-Free Grammar Based Representation. In *CVPR*, 2006.

[54] Adam Sadilek and Henry Kautz. Recognizing Multi-Agent Activities from GPS Data. In *AAAI*, 2010.

[55] W.R. Schwartz, A. Kembhavi, D. Harwood, and L.S. Davis. Human detection using partial least squares analysis. In *ICCV*, 2009.

[56] V.D. Shet, D. Harwood, and L.S. Davis. VidMAP: Video monitoring of activity with prolog. In *AVSS*, pages 224–229, 2005.

[57] Yifan Shi, A. Bobick, and I. Essa. Learning Temporal Sequence Model from Partially Labeled Data. In *CVPR*, 2006.

[58] Hedvig Sidenbladh and Michael J. Black. Learning image statistics for Bayesian tracking. In *ICCV*, 2001.

[59] L. Sigal and M.J. Black. Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In *CVPR*, 2006.

[60] Leonid Sigal, Alexandru O. Balan, and Michael J. Black. HumanEva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *IJCV*, 87(1-2):4–27, 2010.

[61] Michael R. Siracusa and John W. Fisher III. Tractable Bayesian inference of time-series dependence structure. In *AISTATS*, 2009.

[62] Muralikrishna Sridhar, Anthony G Cohn, and David C Hogg. Unsupervised Learning of Event Classes from Video. In *AAAI*, 2010.

[63] Erik B. Sudderth, Michael I. M, William T. Freeman, and Alan S. Willsky. Distributed occlusion reasoning for tracking with nonparametric belief propagation. In *NIPS*, 2004.

[64] Van thinh Vu, Franeois Bremond, and Monique Thonnat. Automatic video interpretation: A novel algorithm for temporal scenario recognition. In *IJCAI*, 2003.

[65] Son D. Tran and Larry S. Davis. Event Modeling and Recognition Using Markov Logic Networks. In *ECCV*, 2008.

[66] R. Urtasun and T. Darrell. Sparse probabilistic regression for activity-independent human pose inference. *CVPR*, 2008.

[67] J. M. Wiener and G. Franz. Isovists as a means to predict spatial experience and behavior. *Lecture notes in artificial intelligence*, 3343:42–57, 02 2005.

[68] Chen Yanover and Yair Weiss. Finding the m most probable configurations using loopy belief propagation. In *NIPS*, 2004.

[69] Ting Yu and Ying Wu. Decentralized multiple target tracking using netted collaborative autonomous trackers. In *CVPR*, 2005.

[70] Tao Zhao and R. Nevatia. Tracking multiple humans in complex situations. *PAMI*, 2004.

[71] Guangyu Zhu, Qingming Huang, Changsheng Xu, Yong Rui, Shuqiang Jiang, Wen Gao, and Hongxun Yao. Trajectory based event tactics analysis in broadcast sports video. In *MULTIMEDIA*, 2007.

[72] Long Zhu, Yuanhao Chen, Yifei Lu, Chenxi Lin, and A. Yuille. Max margin AND/OR graph learning for parsing the human body. In *CVPR*, 2008.