# Tracking Multiple Objects Through Occlusions

Yan Huang, Irfan Essa

Georgia Institute of Technology
GVU Center / College of Computing
Atlanta, GA 30332-0280 USA
{huangy, irfan}@cc.gatech.edu

## Abstract

*We present an approach for tracking varying number of objects through both temporally and spatially significant occlusions. Our method builds on the idea of object permanence to reason about occlusion. To this end, tracking is performed at both the region level and the object level. At the region level, a customized Genetic Algorithm is used to search for optimal region tracks. This limits the scope of object trajectories. At the object level, each object is located based on adaptive appearance models, spatial distributions and inter-occlusion relationships. The proposed architecture is capable of tracking objects even in the presence of long periods of full occlusions. We demonstrate the viability of this approach by experimenting on several videos of a user interacting with a variety of objects on a desktop.*

## 1. Introduction

Visual tracking of multiple objects with continuous interactions is important for many applications that include surveillance and effective user interaction. Complex interactions between objects results in both temporally and spatially significant occlusions, making multi-object tracking a challenging problem. Our goal in this work is to infer and reason about occlusions inherent in multi-object interactions. Using such reasoning allows for visual tracking of a varying number of objects over extended sequences with temporal and spatial occlusions.

Our approach builds on the idea of object permanence [1] to reason about occlusions and presents a *two-level* tracking method that incorporates (i) a region-level association process and (ii) a object-level localization process to track objects through long periods of occlusions. Region association problem is approached as a constrained optimization problem and solved using Genetic Algorithm (GA) [11]. Objects are localized using adaptive appearance models, spatial distributions and occlusion relationships.

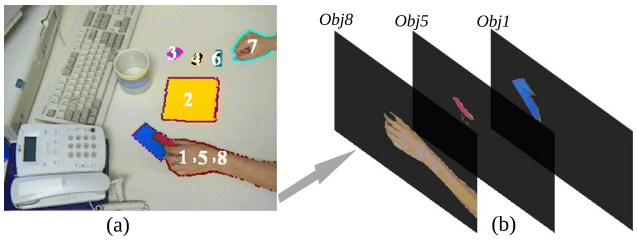


**Figure 1:** Reasoning during occlusions as a user interacts with several objects on the desktop, 6 objects and 2 hands being tracked. (a) shows a frame, where the lower right region consists of three inter-occluding objects: 1, 5, 8. (b) shows the inferred depth ordering among these objects.

In our approach, no assumptions are made about size, shape or motion of objects, only color is assumed to be distinguishable. The prototype system we have built, based on this algorithm is able to perceive the persistence of objects when they are under significant occlusions and also when they re-emerge from behind the occluders. It is also able to discover initially hidden objects when they emerge from existing objects. For testing of our prototype system, we have concentrated on tracking hands and objects as a person interacts with multiple objects on a desktop (Figure 1). Our system can be easily adapted to other domains with further work to relax the requirement of using color as an appearance metric.

We start with a brief overview of existing work in this field that motivates our work and then provide technical details. In section 3 we present the overall architecture of the system and its components followed by descriptions of region and object level tracking in sections 4 and 5. In section 6 we present and discuss our experiments and results.

## 2. Related Work

In earlier research, occlusions is handled either implicitly (without reasoning about occlusions), or explicitly (by inferring depth ordering).

A system that tracks people without explicit reasoning about occlusions, is *Pfinder* [17]. It represents people by a collection of colored regions (blobs). Deletion/addition of blobs from the person model during and after occlusions makes it robust to partial occlusions. Haritaoglu *et al.* [3], McKenna *et al.* [10] and Marque *et al.* [9] use region tracks and appearance model to identify people after occlusions, but they do not compute fine locations of people during occlusions, nor do they support complex interactions. Khan *et al.* [7] maintain a list of persons and classify pixels into them or the background. It can retrieve a person after occlusions, but may break down if he/she has changed color or undergone large displacement. Isard *et al.* [4] present a Bayesian blob-tracker which implicitly handles occlusions by incorporating the number of interacting persons into the observation model and inferring it using Bayesian Network. Jepson *et al.* [5] account for occlusions by an outlier component in a generative appearance model with three components and use online EM to learn the parameters of this model.

Among systems that explicitly reason about occlusions, Wu *et al.* [18] incorporate an extra hidden process for occlusion into a dynamic Bayesian network, and rely on the statistical inference of the hidden process to reveal the occlusion relations. Senior *et al.* [13] use appearance models to localize objects and use disputed pixels to resolve their depth ordering during occlusions. However the system cannot maintain object identity after occlusions. Koller *et al.* [8] track vehicles by using a ground plane constraint to reason about vehicle occlusions. Layer representation has also been used to model occluding objects. Rehg *et al.* [12] represent self-occlusion with layered templates, and use a kinematic model to predict occlusions. Brostow *et al.* [2] automatically decompose video sequences into consitituent layers sorted by depths by combining spatial information with temporal occlusions. Jojic *et al.* [6] and Tao *et al.* [14] both model videos as a layered composition of objects and use EM to infer objects appearances and motions. Zhou *et al.* [19] extend [14] by introducing the concept of background occluding layers and explicitly inferring depth ordering of foreground layers.

Most of the above mentioned approaches rely on partial observations, which makes it difficult to handle full occlusions. In addition, small and consistent motions are assumed that causes problems in dealing with long periods of occlusions under unpredictable motions.

# 3. System Architecture

In order to track multiple objects with complex interactions, it is necessary to perceive the persistence of objects under significant occlusions and when they re-emerge from behind the occluders. We observe that partially or fully occluded objects, even though not observable, still exist in the *close proximity* of their occluders and move with them before they re-emerge, as required by the concept of object permanence. On the other hand, from image processing view of point, objects within close proximity reside in the same connected foreground area, a *region*. Therefore, keeping track of the region in which an object resides limits the scope of feasible object trajectories. For unobservable object, its trajectory can also be estimated based on region tracks. This illustrates the main idea behind our tracking approach: incorporating region-level tracks and object-level tracks to track multiple objects under significant occlusions. Figure 2 shows the structure of our approach. In the first segmentation process, given an image in an input video sequence and a corresponding background image, standard thresholding technique is used to obtain preliminary foreground pixels. These foreground pixels are morphologically closed, smoothed by a median filter, and grouped into regions using connected components algorithm (in its 4-connection neighbors version). A size filter is applied to remove noisy small regions and holes inside remaining regions are filled. The result of this segmentation process is a collection of foreground regions.

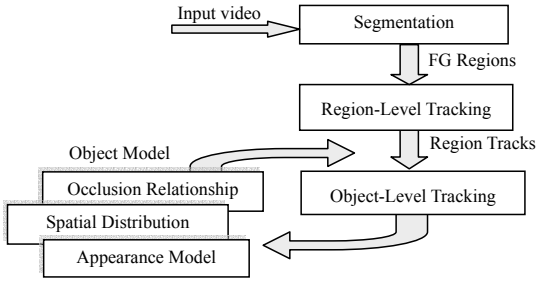


**Figure 2:** Architecture of the tracking system

Due to object proximity to one another, related occlusions and image noise, each foreground region may consist of more than one objects. To construct tracks corresponding to each object, processing is done at two levels: *region-level* and *object-level*.

Firstly, *region-level* tracking procedure associates foreground regions from consecutive frames with each other. Regions may continue, merge, split, appear and disappear, resulting in complex associations. We formulate the region association problem as a global optimization problem and use *Genetic Algorithm* (GA) to search for the optimal conflict-free associations. Resulting region tracks imply the set of objects that can reside in each region, i.e., the *admissible objects* for each region, and therefore, removes from consideration large image areas that do not indicate the presence of an object.

Secondly in each region, *object-level* tracking process lo-

cates objects by registering each individual pixel to one of the admissible objects based on their appearance models and spatial distributions. The appearance model and spatial distribution of an object are initialized once it appears in the scene and gradually updated over time by assembling the registration results. Occlusion relationship among objects in the same region (*neighboring objects*) is also inferred, which helps to locate unobservable objects when the corresponding region splits.

Now we provide further details on region and object level tracking.

# 4. Region-Level Tracking

For each pair of successive frames, the region-level tracking process computes region tracks by associating the new foreground regions (*destination regions*) with the existing regions (*source regions*). These associations can be characterized by a binary *correspondence matrix* $\Theta$ where rows and columns correspond to source and destination regions respectively. An entry is 1 if there is an association between the corresponding regions and 0 otherwise. As a region evolves, the following association events might occur:

**Continuation.** A region continues from the frame at $t-1$ to $t$. The corresponding column and row in $\Theta$ have only one non-zero element.

**Appearing.** A new region appears in the frame at $t$. The corresponding column has all zero elements.

**Disappearing.** An existing region in the frame at $t-1$ disappears. The corresponding row has all zero elements.

**Merging.** Two or more regions in the frame at $t-1$ merge into one region in the frame at $t$. The corresponding column has more than one non-zero entries.

**Splitting.** One region in the frame at $t-1$ separates into two or more regions in the frame at $t$. The corresponding row has more than one non-zero entries.

Let $\gamma$ be the number of association events at time $t$, then a correspondence matrix $\Theta_t$ encodes all these events $E^i_{\Theta_t}(i = 1 \dots \gamma)$, each associating a set of source regions $R^i_{t-1}$ in the previous frame with a set of destination regions $R^i_t$ in the current frame. $R^i_{t-1}$ and $R^i_t$ may each include zero, one or more regions. We also assume that a region cannot be involved in merging and splitting events simultaneously, i.e., compound events are not allowed. Thereby, $R^i_{t-1}$ and $R^i_t$ should not include more than one regions at the same time.

Region tracking problem can therefore be reduced to the problem of determining the correspondence matrix. It's very important to construct the correct correspondence matrix to avoid rejecting correct object tracks in the next object-level tracking process. Methods [16, 9, 13, 10] that form correspondence matrix by thresholding some distance matrix between regions are simple but sensitive to threshold and might fail if regions exhibit different sizes, arbitrary shapes and significant motions.

We approach the problems of reliably constructing a correspondence matrix as a constrained optimization problem. That is, we search for the optimal correspondence matrix $\Theta^*_t$ in the feasible space to maximize the dimensionless global posterior of the matrix conditioned on observed measurements $o_{t-1}, o_t$ (segmented foreground regions). Assuming uniform prior, we have:

$$\Theta^*_t = \arg\max_{\Theta_t} p(\Theta_t|o_{t-1}, o_t) \propto \arg\max_{\Theta_t} p(o_{t-1}, o_t|\Theta_t),$$

$$\Theta^*_t = \arg\max_{\Theta_t} \prod_{i=1}^{k} p(R^i_{t-1}, R^i_t|E^i_{\Theta_t}) \quad (1)$$

where $E^i_{\Theta_t}, i = 1 \dots k$ are events encoded by $\Theta_t$, $R^i_{t-1}$ and $R^i_t$ are sets of regions associated by the event $E^i_{\Theta_t}$, $p(R^i_{t-1}, R^i_t|E^i_{\Theta_t})$ is the likelihood that regions $R^i_t$ come from regions $R^i_{t-1}$ .

Now we discuss the key issues to find $\Theta^*_t$: evaluating correspondence matrix and searching for the optimal one.

## 4.1. Evaluating Correspondence Matrix

Given an event $E^i_{\Theta_t}$, the likelihood that the corresponding regions $R^i_t$ come from regions $R^i_{t-1}$ can be computed in one of the following ways:

$$p(R^i_{t-1}, R^i_t|E^i_{\Theta_t}) = \begin{cases} p_1(R^i_{t-1}) & \text{if } |R^i_t| = 0\text{(disappearing)} \\ p_1(R^i_t) & \text{if } |R^i_{t-1}| = 0\text{(appearing)} \\ p_2(R^i_{t-1}, R^i_t) & \text{otherwise} \end{cases} \quad (2)$$

where $|R^i|$ is the number of regions in the set $R^i$. If $E^i_{\Theta_t}$ is a disappearing event, region $R^i_{t-1}$ associates with no destination regions, i.e., $|R^i_t| = 0$. Similarly, if $E^i_{\Theta_t}$ is an appearing event, $|R^i_{t-1}| = 0$. Appearing /disappearing likelihood $p_1$ of a region is a predefined Gaussian density function which increases as the region size and the distance from the region centroid to the image boundary decrease.

In the cases of continuation, merging and splitting events, the likelihood that $R^i_t$ comes from $R^i_{t-1}$ is computed by combining their displacement probability $p_{\text{dis}}$ and shape similarity $p_{\text{shape}}$:

$$p_2(R^i_{t-1}, R^i_t) = p_{\text{dis}}(R^i_{t-1}, R^i_t) \cdot p_{\text{shape}}(R^i_{t-1}, R^i_t) \quad (3)$$

where $p_{\text{dis}}$ is a predefined Gaussian density function that increases as the distance between the region centroids decreases. *Shape matrix* [15] $SM$ is employed to model shape

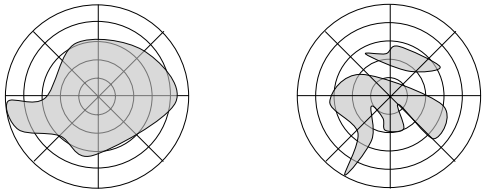

**Figure 3:** Using shape matrix to model regions. (a) Modelling of one region. (b)Modelling of multiple regions.

and compute shape similarity $p_{\text{shape}}$ between two (sets) of regions. Shape matrix is a binary matrix that represents the pixels corresponding to a polar raster of coordinates centered in the region centroid, as illustrated in Figure 3(a). A *shape vector* $SV$ that represents the relative area of the shape contained in concentric rings is then computed by $SV(i) = \frac{\sum_j SM(i,j)}{\sum_i \sum_j SM(i,j)}$. In the case of continuing event, there is one source region and one destination region. A shape vector can be obtained for each of them. In the case of merging or splitting event, there are either more than one source regions or more than one destination regions. Therefore we need to model multiple regions with one shape vector, as shown in Figure 3(b). The process is essentially the same as that for a single region, except that the raster is centered in the centroid of all regions. Given two sets of regions, the similarity between their shapes is given by the Bhattacharyya coefficient of their shape vectors:

$$p_{\text{shape}}(R^i_{t-1}, R^i_t) = \sum_{u=1}^{m} \sqrt{(SV^i_{t-1})_u \cdot (SV^i_t)_u} \qquad (4)$$

where $SV^i_{t-1}$ and $SV^i_t$ are the shape vectors of $R^i_{t-1}$ and $R^i_t$ respectively, $m$ is the dimension of each shape vector.

## 4.2. Randomized Optimization

Equation (1) indicates that, given a correspondence matrix hypothesis, its global likelihood is closely related to the likelihood of each encoded association event. This implies the optimal association event for one region is highly likely to be part of the overall optimal solution. This is analogous to the crucial presumption in Genetic Algorithm – *building block*, which is strongly related to the high values of the evaluation function. Genetic algorithm is a randomized searching algorithm that can search for spaces of hypotheses containing complex interacting parts to identify the best hypothesis. Therefore, it fits our optimization problem structure well.

To apply GA for searching, we start by initializing a population of hypotheses in the feasible space near the optimal solution. We then proceed by iteratively updating the population until termination conditions are met. On each iteration, individuals in the population are evaluated using Equation (2). A new generation is then generated by probabilistically selecting the most fit individuals from the current population. These are used as the basis for creating new offspring individuals by applying genetic operators such as crossover and mutation.

We design the initialization, crossover and mutation operations based on three guidelines: (1) Enforcing all created individuals to stay in the feasible hypotheses space, e.g, a region cannot be involved in *compound events*. (2) Leveraging heuristics for correct jump and faster convergency, e.g., in crossover operation, combining promising building blocks of parents to create child. (3) Introducing randomness to guarantee the whole feasible space is reachable.

**Initialization:** To initialize a number of individuals that cover the desirable subspace while ensuring sufficient diversity, some heuristics are used. A destination region is more likely to be associated with source regions that are spatially close to it. Therefore, for each destination region, we probabilistically initialize its association source regions based on their bounding box distance [13]: the smaller the distance, the more probable to be selected. If the addition of a selected association may result in an infeasible individual, the destination region is randomly associated to a source region that will not break constraints. Moreover, to increase population diversity, we initialize individuals by scheduling the assignment of their destination regions following different permutation orders. Therefore, if $m$ is the number of destination regions, there would be at most $m!$ number of assignment orders to follow.

**Crossover:** A parent may consist of both good and bad association events. Given a pool of mating parents, it is desirable to have parents with different strong points to mate in order to create a child that dominates both parents. Therefore parents in the mating pool are grouped according to their best association events. To produce a child, a given parent is more likely to mate with a parent from a different group. During mating process, instead of random multi-point crossover, we utilize building block crossover, namely, as many good association events as possible from parents are inherited by their child before constraints are broken. The remaining destination regions are randomly associated with source regions without breaking constraints.

**Mutation:** Mutation is activated with a small probability. For a random destination region in a randomly selected individual, we perform *local search* to find a better local event for the given destination region. For the matrix column corresponding to the destination region, We create new events by flipping one entry at a time if constraints are not broken. A change is saved if it results in better likelihood.

# 5. Object-Level Tracking

Region level tracks indicate the admissible object labels for regions in the current frame. Let $L^j_{t-1}$ be the set of object labels for the source region $r^j_{t-1}$. Given the region tracks, we can derive $L^i_t$, the set of object labels for the destination region $r^i_t$, based on the corresponding association events listed below.

**Appearing.** $L^i_t = \{l_{\text{new}}\}$, where $l_{\text{new}}$ is a newly assigned object label. Here we assume that every time a new object appears to the scene, it is isolated. Namely, it is the only object inside the corresponding region. This applies to all appearing regions in the sequence as well as those objects that were initially hidden but then separated out forming a new region in the mid of the sequence.

**Disappearing.** The region does not exist any more, so there is no need to assign labels.

**Continuation and Merging.** $L^i_t = \bigcup_{j \in I_i} L^j_{t-1}$, where $I_i$ denotes the set of indices of the corresponding source regions.

**Splitting.** Each destination region may either consist of a subset of the objects from the source region, or represent a new object which has been hidden so far and then separated out. Therefore, $\bigcup_{i \in I_j} L^i_t = L^j_{t-1} \cup L_{\text{new}}$, and $\forall a \neq b (a, b \in I_j), L^a_t \cap L^b_t = \phi$, where $I_j$ denotes the set of indices of the corresponding destination regions, $L_{\text{new}}$ denotes a set of zero, one or more new objects. In this case, unlike all other cases, there is no unique assignment of labels for each destination region and we must compute the most probable one.

In order to localize and track each object inside a region consistently, we need a robust model for each object.

## 5.1. Object Model

We model an object by its *appearance model*, *spatial distribution* as well as the *occlusion relationships* with its neighboring objects. In our prototype system, the appearance model is a color histogram built over H and S channels in HSV color space. H channel is quantized into 16 values and S into 8 values.

However, color histogram lacks spatial layout information and may fail to characterize objects when they have similar colors, which can sometimes be solved by incorporating the spatial distribution of the object area. Spatial distribution is the first and second moments of the 2-D coordinates of the object, i.e., object centroid and full covariance.

Moreover, when a region splits while some of its objects are fully occluded both before and after the splitting event, partial depth ordering between its objects can help to estimate where these objects reside after split. Therefore, we also reason about partial depth ordering as a part of the object model.

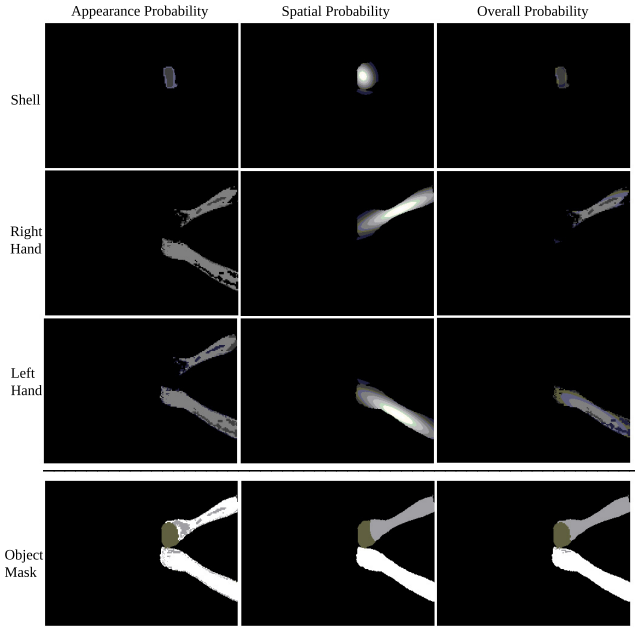


**Figure 4:** Comparison of the registration processes based on pure appearance models (left column), pure spatial distributions (middle column) and their combinations (right column). The source frame is shown in Figure 8, frame 325. The upper part shows probabilities of pixels belonging to each object: white stands for high probability, dark for low probability. The lower part shows the resulting collection of pixels for each object, masked by different colors, registered based on the three probabilities respectively.

Object models are automatically initialized the first time they appear. They are then used to localize objects inside regions, and updated based on the localization results.

## 5.2. Objects Localization

To localize objects inside a destination region, different processes are needed depending on the association event of the region: (1) For non-splitting event, the destination region has unique objects assignment, hence we only need to localize these objects inside the region; (2) For splitting event, there are more than one feasible assignments, so we must find the best one before locating objects accordingly.

**Case 1: Non-Splitting Events** For an appearing region $r^i$, it represents a new object: $L^i_t = \{l_{\text{new}}\}$.

For a continued or merged region, its admissible labels can be directly inferred. Since objects may have arbitrary shapes and may be non-rigid, it is not feasible to locate them as coherent areas. Instead, we register each individual pixel inside the region independently to one of the admissible objects that best claims it. The collection of pixels belonging

to an object will be assembled to derive its location and appearance later on. Pixels are registered based on the appearance models and the spatial distributions of the admissible objects. Let $\{l_i\}(i = 1 \cdots n)$ be the set of $n$ admissible objects, pixel $\mathbf{x}$ in the region should be registered to the object that maximizes the posterior probability:

$$i^* = \arg\max_i p(i|\mathbf{x}) \propto \arg\max_i p(\mathbf{x}|i),$$

$$i^* = \arg\max_i p_{\text{app}}(\mathbf{x}|i) \cdot p_{\text{spa}}(\mathbf{x}|i), \quad (5)$$

where $p_{\text{app}}$ is the appearance likelihood and $p_{\text{spa}}$ is the spatial distribution likelihood that pixel $\mathbf{x}$ comes from object $l_i$. Let $H^i$ be the color histogram of size $U$ for object $l_i$, the appearance likelihood of color index $c_{\mathbf{x}}$ at pixel $\mathbf{x}$ can be obtained by looking up histogram:

$$p_{\text{app}}(\mathbf{x}|i) = \frac{H^i(c_{\mathbf{x}})}{\sum_{u=1}^{U} H^i(u)}.$$

Let $(\mu_i, \Sigma_i)$ denotes the shape distribution of object $l_i$, $p_{\text{spa}}$ can be obtained from Gaussian density $p(\mathbf{x}|\mu_i, \Sigma_i)$. Figure 4 illustrates and compares the registration processes of a region (Figure 8, frame 325) based on appearance models, spatial distributions and their combinations. The upper part visualizes the appearance probabilities, spatial probabilities and overall probabilities of each pixel in the region belonging to the three neighboring objects (shell, left hand and right hand). The lower part shows the corresponding collection of pixels for each object, masked by different colors. As illustrated, the left and right hands exhibit similar colors and cannot be correctly registered based on their appearance models. Part of the pixels belonging to the shell are registered to the right hand if only the estimated spatial distributions are used. However, by compensating appearance models with spatial distributions, we can obtain much better registration results.

Once we have the collection of pixels belonging to object $l_i$, we can aggregate them to obtain the following properties : (1) occlusion ratio or visibility by comparing the new object size with the original size; (2) new appearance model $\widetilde{H_t^i}$ by counting colors; (3) new centroid and covariance from the coordinates of the collection of pixels. With these properties we can then update object models. However, due to the inevitable noises brought by the independent pixel-level registration process, we slowly blend the new information into the existing models: *(A) Appearance model*: new appearance model is gradually blended into the old appearance with an update form:

$$H_t^i = \beta \cdot H_{t-1}^i + (1 - \beta) \cdot \widetilde{H_t^i} \quad (6)$$

where $\beta = 0.9$. The adaption of appearance model is suspended if the object is in substantial occlusions. *(B)Spatial distribution*: if the object is not substantially occluded, update its spacial distribution with the new one; otherwise, we assume the object moves with its region and so offset its centroid with the displacement of the corresponding region. *(C) Occlusion relationship*: if the object is sufficiently occluded, its occluders are estimated. Its former occluders are inherited. Meanwhile, new occluders are inferred using a simple heuristic: an object is an occluder of the occludee if it occupies sufficiently large areas that belonged to the occludee in the last frame. If no occluder is discovered, we assign its neighboring objects as occluders. This heuristic works for most of the situation in our experiments, but may fail sometimes due to the noisy pixel registration results. However, it will recover from the error when registration improves later on.

Once we have pairwise occlusion relationship between neighboring objects, we can infer their partial depth ordering, or layers, as illustrated in Figure 1. Figure 1(a) shows a frame whose lower right region consists of object 1(blue block), 5(red block) and 8(hand), where the hand partially occludes the red block and the red block partially occludes the blue one. These occlusion relationships are correctly inferred and their depth ordering is displayed in (b). The partial occlusion relationships are continuously estimated. When full occlusions take place, the layer information can still be estimated accordingly. Layer information plays an important role for estimating where a fully occluded object resides after region splits, especially for our shell game experiment where this happens all the time.

**Case 2: Splitting Events** For split, we have more than one way to assign labels to destination regions. Let $n$ be the number of objects inherited from the source region, $m$ be the number of destination regions from split. Given the constraints $\bigcup_{i \in Ij} L_t^i = L_{t-1}^j \cup L_{\text{new}}$ and $\forall a \neq b (a, b \in I_j), L_t^a \cap L_t^b = \phi$, there are altogether $m^n$ number of feasible assignments. We want to find from them the one that is most probable. Considering that $n$ and $m$ are typically small, it is affordable to enumerate all of the assignments, evaluate them, and find the best one. An assignment is evaluated by computing how likely each object exists in the assigned region.

Let $\Phi = (\Phi^1, \Phi^2, \cdots, \Phi^m)$ be an assignment for the $m$ regions, where $\Phi^i = (l_1^i, l_2^i, \cdots, l_{k_i}^i)$ denotes the set of $k_i$ objects assigned to region $r^i$. The likelihood of $\Phi$ is the product of the likelihood that $r^i$ represents $\Phi^i$:

$$p(o_t|\Phi) = \prod_{i=1}^{m} p(r^i|\Phi^i) \quad (7)$$

If region $r^i$ is assigned to a new object ($\Phi^i \subseteq L_{\text{new}}$), it represents an initially hidden object, then $p(r^i|\Phi^i)$ is a predefined constant; otherwise we register pixels in $r^i$ with objects $\Phi^i$ following the same procedure as that in case 1.

Let $r_j^i (j = 1, \cdots, k_i)$ be the collection of pixels registered to object $l_j^i$. By deriving from $r_j^i$ the observed appearance $\widetilde{H^{l_j^i}}$, size, centroid and occlusion relationship for object $l_j^i$, we can compute the likelihood that $r_j^i$ represents $l_j^i$ by:

$$p(r_j^i | l_j^i) = p_{\text{h}} \cdot p_{\text{size}} \cdot p_{\text{dis}} \cdot p_{\text{occ}} \tag{8}$$

where $p_{\text{h}}$ is the similarity between observed histogram $\widetilde{H^{l_j^i}}$ and object histogram model $H^{l_j^i}$ computed as their Bhattacharyya coefficient, $p_{\text{size}}$ and $p_{\text{dis}}$ are size likelihood and centroid distance likelihood respectively which are both predefined Gaussian density functions that decrease as size difference and centroid distance increase, $p_{\text{occ}}$ is occlusion likelihood which awards the occlusion relationship consistent with that from previous frame and penalizes confliction. With $p(r_j^i | l_j^i)$ for each object in each region, we can evaluate the association $\Phi$ as:

$$p(o_t | \Phi) = \prod_{i=1}^{m} p(r^i | \Phi^i) = \prod_{i=1}^{m} \prod_{j=1}^{k^i} p(r_j^i | l_j^i) \tag{9}$$

# 6. Results

We have experimented with 5 indoor video sequences and present the results of two representative ones: lego sequence and shell game (see included videos). Our approach worked equally well for all of our test sequences. All sequences are 320*240 color videos captured by a single fixed camera and digitized at 15 fps. All of them have varying number of objects with unconstrained sizes, shapes and motions, but with relatively simple colors and limited textures. Temporally and spatially significant occlusions are present throughout all sequences.

In all our experiments, a background image was captured for segmentation purposes. Limited manual intervention was required in the segmentation process to avoid compound association events at region level.

**Lego Sequence (412 frames):** This is a sequence where an actor moves a number of lego blocks around a desk, stacks and unstacks them, and removes them from the scene. Some blocks have similar colors. Segmentation of 2 out of 412 frames resulted in compound region association events and were manually fixed by changing segmentation threshold. Figure 5 is a visual depiction of the interactions among the 10 objects tracked in the sequence. Each line represents an object. Lines merge when the corresponding objects move into the same region, and split when the objects separate. Figure 6 shows some selected frames in the sequence. The upper row shows the frames imposed by the tracked object IDs. The lower row is the corresponding

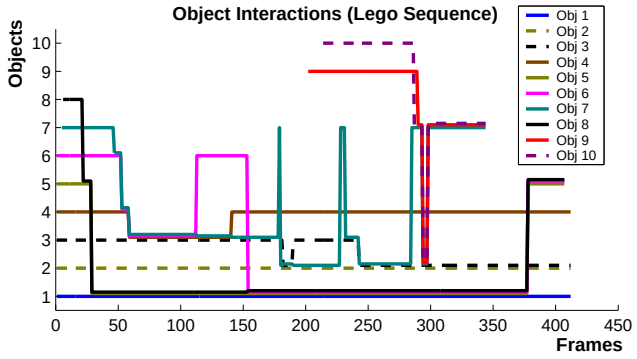


**Figure 5:** Interactions among the 10 objects in the lego sequence. X and Y axis represent time and object index respectively. Each line represents an object. Lines merge when the corresponding objects move into the same region, and split when the objects separate.

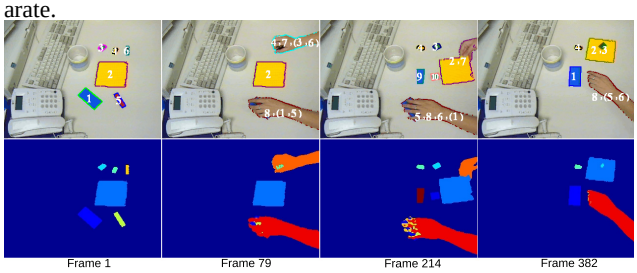


**Figure 6:** Selected frames in the lego sequence.The upper row shows the frames imposed by the tracked object IDs. IDs inside brackets stand for significantly occluded objects. The lower row is the corresponding mask of each object distinguished by different colors. See included video.

mask of each object distinguished by different colors. Notice that during the sequence, almost all blocks were significantly occluded for tens of continuous frames. Particularly, object 5 was fully occluded by the left hand and meanwhile moving with it with non-uniform motions for as long as 357 continuous frames and as much as 186 displacement in pixels. Two blocks $\{9, 10\}$ initially hidden by the yellow block $\{2\}$ were discovered and tracked. Moreover, the size and shape of the two hands (arms) changed dramatically during the sequence. Despite all these difficulties, our algorithm is able to consistently track all objects.

**Shell Game Sequence (460 frames):** This is a sequence of a variant on the popular "shell game." An actor starts by slipping two chips of different colors underneath three identical shells (cups) face down on a table and slides the shells around for an extended time before uncovering them to expose chips. Two chips, rather than one, as in a typical shell game, were used to make it more difficult for human to remember where each chip goes. Chips are of different colors in order to make it possible to tell at the end of sequence

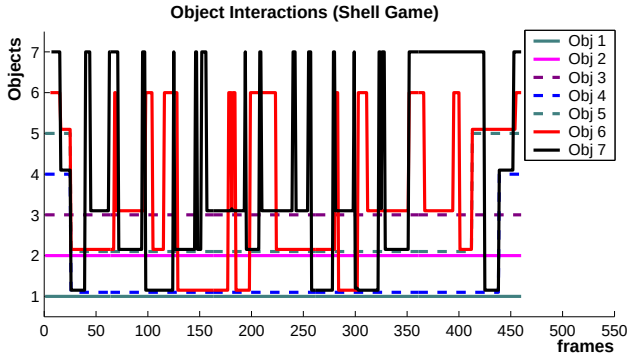


**Figure 7:** Interactions among the 7 objects in the shell game sequence.

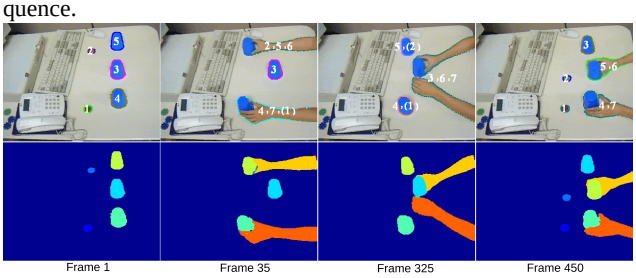


**Figure 8:** Selected frames in the shell game sequence. See included video.

whether the tracking system is successful at "guessing". Only one frame out of 460 frames resulted in compound region association event and were manually fixed. Figure 7 visualizes the interactions among the 7 objects tracked in the sequence. As shown, objects exhibit repeated interaction patterns due to the nature of shell game. Figure 8 illustrates some selected frames. Again, all objects were successfully tracked throughout the sequence, and correct correspondence between chips and shells were given by the tracker.

In both sequences, some pixels in some frames were not correctly registered, as shown in the object masks part of Figure 6 and 8, due to the shadows of hands and shells. However, in most cases this did not prevent the tracker from correctly estimating object trajectories and reasoning about occlusion relationships. We only found a few frames where occlusion relationships were not correctly predicted, but due to the incorporation of appearance model and spatial distribution in trajectory computation, all objects were still correctly located.

## 7. Conclusions and Future Work

This paper presents an approach and a prototype system for tracking varying number of objects through significant occlusions without specific assumptions about their appearances, shapes or motions. The use of the two-level tracking approach and the adaptive object models enable the proposed system to successfully track objects even in the presence of significant occlusions. Our approach is also able to discover and track initially hidden objects as they emerge from behind existing objects.

Our future work includes the exploration of more robust appearance models in order to track objects with complex textures. We also plan to use adaptive background models to handle varying illumination conditions.

## References

[1] R. Baillargeon, E. Spelke, and S. Wasserman. Object permanence in five-month-old infants. *Cognition*, 80:191–208, 1985.

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

[3] I. Haritaoglu, D. Harwood, and L. Davis. W4: Who? When? Where? What? a real time system for detecting and tracking people. pages 222–227, 1998.

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

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

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

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

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

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

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

[11] T. Mitchell. *Machine Learning*. McGraw Hill, New York, 1997.

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

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

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

[15] A. Taza and C.Y. Suen. Discrimination of planar shapes using shape matrices. *IEEE Transactions on Systems, Man, and Cybernetics*, 19:1281–1289, 1989.

[16] J. A. Withers and K. A. Robbins. Tracking cell splits and merges. In *IEEE Southwest Symposium on Image Analysis and Interpretation*, 1996.

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

[18] Y. Wu, T. Yu, and G. Hua. Tracking appearances with occlusions. In *CVPR*, 2003.

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