Human Tracking with Mixtures of Trees

Paper ID: 629

Abstract

Tree-structured probabilistic models admit simple, fast inference. However, they are not well suited to phenomena such as occlusion, where multiple components of an object may disappear simultaneously. Mixtures of trees appear to address this problem, at the cost of representing a large mixture. We demonstrate an efficient and compact representation of this mixture, which admits simple learning and inference algorithms.

We use this method to build an automated tracker for Muybridge sequences of a variety of human activities. Tracking is difficult, because the temporal dependencies rule out simple inference methods. We show how to use our model for efficient inference, using a method that employs alternate spatial and temporal inference. The result is a tracker that (a) uses a very loose motion model, and so can track many different activities at a variable frame rate and (b) is entirely automatic.

1. Introduction

One of the main difficulties in object recognition is the variability of appearance of an object of interest. The reasons for these variations include the articulated nature of an object, variations in aspect, and occlusions. However, articulated objects can be represented as assemblies of rigid parts. For example, a human body is often represented as an assembly of cylinders which move with respect to each other. Such a representation suggests a bottom-up approach to recognition: first, we identify *candidate primitives* as image regions that may correspond to the object parts; then, these regions are grouped into *assemblies* that satisfy constraints on the relative configuration of the parts. If the object is known to be in the image, it can be localized by finding the assembly that is the most similar to the object of interest.

We cannot evaluate each group of candidate primitives due to the large number of such groups, and thus need an efficient grouping method. Such a method is provided by tree-structured probabilistic models which admit simple, fast inference [2, 7]. However, they are not well suited to phenomena such as occlusion and aspect variation, where multiple components of an object may disappear simultaneously. This problem can be addressed with *mixtures of trees* [3], with each component modeling a particular aspect. By imposing constraints on the structure of mixture components, we can represent the large number of components efficiently and compactly. Such a representation admits simple learning and inference algorithms.

We use mixtures of trees to model people, and track them in Muybridge sequences of a variety of human activities. Tracking is difficult, because the temporal dependencies rule out simple inference methods. The problem is often simplified by imposing tight motion models and/or manual initialization [1, 6], neither of which can be used in a fully automatic tracker on Muybridge sequences, which have low and variable frame rate. Instead, we combine the mixture-of-trees model for a human body with a weak motion model, and show how to use our model to search for human configurations in motion sequences efficiently, using a method that employs alternate spatial and temporal inference. The result is a tracker that (a) uses a very loose motion model, and so can track many different activities at a variable frame rate and (b) is entirely automatic.

2. Modeling with trees

Consider an object that is formed from a set of primitives. We will detect such an object by first detecting the primitives, and then grouping them into assemblies. For instance, we can look for people as collections of body parts. There is little hope of reliably detecting individual primitives without looking at a configuration as a whole. Instead, we propose not to find the primitives accurately, but rather find a set of possible configurations for each primitive, and use the grouping process to determine which element of a candidate set is in fact a part of an object. For instance, we can find several image regions that could correspond to an arm in the image of a person, and use grouping to determine which of those in fact corresponds to an arm – perhaps, a region that is adjacent to something that looks like a torso.

Let us suppose that an object is a collection of K primitives, $\{X_1 \dots X_K\}$, each of which can be treated as a vector representing its *configuration* (e.g., position and orientation in the image). Given an image, the local detectors will provide us with a finite set of possible configurations for each primitive X_k. We will refer to each of these K sets as a set of *candidate primitives*; the objective is to build an *assembly* by choosing an element from each candidate set, so that the resulting set of primitives satisfies some global constraints (e.g., limbs must be attached to the torso).

The global constraints can be captured in a distribution $P(X_1 \dots X_K)$, which will be high when the assembly looks like the object of interest, and low when it doesn't. Assuming exactly one object present in the image, we can localize the object by picking the configuration for each primitive from the corresponding candidate set so that the

resulting value of P is maximized. In general, this maximization requires a combinatorial correspondence search whose computational cost is prohibitively large (unless K is small, e.g. [10]). It is possible, however, to constrain ourselves to a family of distributions P for which correspondence search is efficient. This is the case if P corresponds to a tree-structured graphical model.

If $P(X_1 \dots X_K)$ is represented with a tree, correspondence search is efficiently accomplished with a Viterbi algorithm whereby the nodes of the tree are swept from leaves to the root. If there are M candidate configurations for each of the K primitives, then the search takes $O(KM^2)$ time, whereas for a general distribution P the complexity would be $O(M^K)$.

3. Learning the tree model

In addition to making correspondence search efficient, the conditional independences captured in the tree model simplify learning, by reducing the number of parameters to be estimated, due to the factorized form of the distribution:

$$P(X_1 \dots X_K) = P(X_{\text{root}}) \prod_{k \neq \text{root}} P(X_k \mid \text{Pa}_k),$$

where X_{root} is the node at the root of the tree, and Pa_k denotes the parent of the node X_k. Learning the model involves learning the structure (i.e., the tree edges) as well as the parameters of the prior $P(X_{\text{root}})$ and conditionals $P(X_k \mid \text{Pa}_k)$. We learn the model by maximizing the log-likelihood of the training data $\sum_{n=1}^{N} \log P(x_1^n, \dots, x_K^n) = \left(\sum_{n=1}^{N} \log P(x_{\text{root}}^n)\right) + \sum_{k=1}^{K} \left(\sum_{n=1}^{N} \log P(x_k^n \mid \text{pa}_k^n)\right)$ where x_k^n and pa_i^n are the configurations of the kth primitive (X_k) and its parent (Pa_i) in the nth training assembly. If the structure of the tree is known, the terms within parentheses can be maximized independently of each other, yielding the maximum-likelihood approximations for the prior and conditionals. If we now assume that these approximations are in fact the *true* prior and conditionals, then the expected value of the log-likelihood is

$$-N[H(X_{\text{root}}) + \sum_{k \neq \text{root}} H(X_k \mid \text{Pa}_k)],$$

where $H(X_{\text{root}})$ and $H(X_k \mid \text{Pa}_k)$ denote entropy and conditional entropy, respectively. We seek the tree structure that maximizes the log-likelihood of the data, or, equivalently, minimizes the entropy of the distribution whose marginals are constrained by the data.

To learn the tree structure with a fixed root X_{root}, we learn the conditionals $P(X_k \mid \text{Pa}_k)$ and then find the minimum spanning tree in the directed graph, whose edge weights are the appropriate conditional entropies. This tree can be found efficiently [8].

4. Mixtures of trees

The tree representation of an object allows for efficient learning and search. However, it is difficult to use a tree to model cases where some of the primitives constituting an object are missing – due to occlusions, variations in aspect or failures of the local detectors. One approach is to marginalize over missing correspondences [7], so that a conditional $P(X_k \mid \text{Pa}_k)$ is set to an appropriate constant if X_k and/or Pa_k is absent. However, in this case a missing primitive will "break" the assembly, by rendering the descendants of a missing primitive independent from the rest of the assembly. Furthermore, this solution ignores the fact that the presences or absences of primitives influence each other (for example, if an upper arm segment is missing, it is often due to the pose of a person, which will cause the lower arm segment to be absent as well).

Mixtures of trees, introduced in [3], provide an alternative solution. In particular, we can think of assemblies as being generated by a mixture model, whose class variable specifies what set S of primitives will constitute an object, while conditional class distributions $P_S(\{X_k : k \in S\})$ generate the configurations of those primitives. The mixture distribution is

$$P(\{X_k : k \in S\}) = \pi(S) P_S(\{X_k : k \in S\}),$$

where $\pi(S)$ is the probability that a random view of an object consists of those primitives. This mixture has 2^K components – one for each possible subset S of primitive types. Learning a mixture of trees involves estimating the mixture weights $\pi(S)$, as well the structure and the distributions $P_S(X_{\text{root}})$ and $P_S(X_k \mid \text{Pa}_k)$ for each of the component trees.

In [3], tree mixtures are learned using an EM algorithm, the M-step involving optimization of the tree structure and distribution parameters. We cannot take this approach, however: since we require a component for each set of primitives constituting an object, the mixture has 2^K components and, if each component is represented explicitly, cannot be learned or even represented unless K is small. One solution might be to select a small number of mixture components (or subsets S of primitives) that adequately represent the variations in the object appearance. Instead, we keep all of the 2^K components, but represent them implicitly and compactly. To do this and to make learning and inference efficient, we must constrain the structure of the mixture and thus enforce conditional independences similar to those present in a single tree.

5. Trees with shared structure

In a tree model, fixing a value of a node renders its descendants conditionally independent from the other nodes in the model. This property makes learning and the correspondence on trees efficient; to achieve similar efficiency, and simply to be able to represent the huge number of mixture components, we require our model to possess similar conditional independence structure.

In particular, if we specify the configuration of a primitive X_k and assert that X_k is a part of the object, then we want the set of primitives $X_1 \dots X_K$ to break into groups

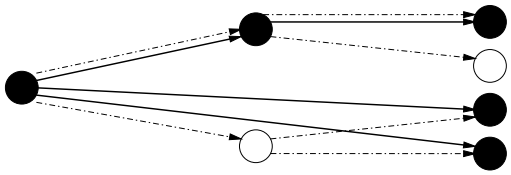


Figure 1: *Using a generating tree to derive the structure for a mixture component. The dashed lines are the edges in the generating tree, which spans all of the nodes. The nodes of the mixture component are shaded, and its edges (shown as solid) are obtained by making a grandparent "adopt" a node if its parent is not present in this tree (i.e., is not shaded). Thus mixture components are encoded implicitly, which allows efficient representation, learning and inference for mixtures with a large number of components.*

so that primitives in different groups are conditionally independent. To achieve this, we use a single *generating tree* which is used to generate the structures of all of the mixture components.

A *generating tree* is a directed tree T whose nodes are $X_1 \ldots X_K$, with $X_{\rm root}$ at the root. For a given structure of T, we learn the prior $P(X_{\rm root})$ and conditionals $P(X_k \mid X_j)$ where X_j is any ancestor of X_k (not necessarily the parent). Additionally, we denote by $[X_k]$ the event that X_k is one of the primitives constituting a random view of the object, and learn the distributions $P([X_{\rm root}])$ and $P([X_k] \mid [Pa_k])$, where Pa_k is the parent of X_k in the generating tree.

We represent $\pi(S)$ using a graphical model with structure provided by the generating tree T:

$$\pi(S) = P([X_{\rm root}]) \prod_{k \neq {\rm root}} P([X_k] \mid [Pa_k]),$$

where the distributions are learned by counting occurrences of each primitive and pairs of primitives in the training data.

The tree P_S consists of all the edges $(X_j \rightarrow X_k)$ such that X_j is an ancestor of X_k in the generating tree, and none of the nodes on the path from X_j to X_k is in the set $\{X_k : k \in S\}$. This means that, if the parent of node X_k is not present in a view of the object, then X_k is "adopted" by its grandparent, or, if that one is absent as well, a great-grandparent, etc. If we assume that the root $X_{\rm root}$ is *always* a part of the object, then P_S will be a tree, since $X_{\rm root}$ will ensure that the graphical model is connected. An example of obtaining the structure of a mixture component is shown in figure 1.

The distribution P_S is the product of the prior $P_S(X_{\rm root})$ and conditionals $P_S(X_k \mid X_j)$ corresponding to the tree edges. To ensure conditional independences in the mixture, we require that if an edge $(X_j \rightarrow X_k)$ is present in several mixture components then the corresponding conditionals are the same; similarly, we require that $P_S(X_{\rm root})$ be independent of S. To this end, we set $P_S(X_k \mid X_j) = P(X_k \mid X_j)$ to be the conditional distribution learned from all of the data, and similar for $P_S(X_{\rm root}) = P(X_{\rm root})$.

As in the case of a single tree, we need to determine the optimal structure for the mixture components. Since they are all derived from a single generating tree T, only its structure needs to be learned, and we do that by minimizing the entropy of the mixture distribution. Therefore, we prefer distributions that are more concentrated, and dislike less informative ones, which have high entropy.

Let us use the notation $[X_j \rightarrow X_k]$ to denote the event that a mixture component contains the edge $(X_j \rightarrow X_k)$ (for a fixed T, the probability of this is a product of marginals; for instance, if X_j is X_k's grandparent then $\Pr([X_j \rightarrow X_k]) = \Pr([X_j]) \cdot \Pr(\overline{[Pa_k]} \mid [X_j]) \cdot \Pr([X_k] \mid \overline{[Pa_k]})$). Then, the entropy of the mixture is

$$\begin{aligned} H(P) &= \sum_{k=1}^{K} H([X_k] \mid [X_j]) \\ &+ \sum_{k,j} \sum \Pr([X_j \rightarrow X_k]) \cdot H(X_k \mid X_j), \end{aligned}$$

where the first term on the right-hand side represents the amount of information that the mixture model P captures about the presence or absence of primitives, and the second term represents the amount of information that is captured about the relative configurations of pairs of primitives.

Unlike the case of a single-tree case, where the optimal structure is obtained by solving the minimum spanning tree problem, in the case of a mixture we are not aware of any efficient algorithms guaranteed to yield the optimal solution. Instead, we use a greedy algorithm that starts with an initial structure of T and repeatedly applies local mutations (such as reassigning a node to a different parent), each of which increases the entropy, and stops when no further increase is possible.

6. Grouping using mixtures of trees

Assuming that exactly one object is present in the image, we need to localize it – that is, select a subset of primitive types and the configurations for those primitives so that the resulting assembly is likely to appear in a random view of the object, but not in a random view of something else. Thus, we find the assembly for which the posterior $\Pr(\text{object} \mid \text{assembly})$ is the largest. Maximizing this posterior is equivalent to maximizing the Bayes Factor $B = P(\{X_k\})/P_{neg}(\{X_k\})$, where the likelihood in the numerator is the probability of a configuration in a random view of the object, and the denominator is the probability of seeing it in the background. Conditional independence structure of our tree mixture model, combined with an appropriately chosen model of the background, makes Bayes factor maximization efficient.

We model background as a collection of independent primitives: the number of primitives in a non-object has a

geometric prior, the type of a primitive is k with probability β_k, and the distribution on the configuration of each primitive is uniform in the region of interest and is common to all of the primitives. Therefore, $P_{neg}(\{X_k : k \in S\}) = \prod_{k \in S}(\alpha\beta_k)$ where β_k is found as the fraction of a particular type of primitive among all the candidates in images of non-objects, and α is the parameter that is estimated so that the classification error is minimized.

Because of the independent structure of P_{neg}, the Bayes factor can be obtained by associating the term $(\alpha\beta_k)^{-1}$ with each member of X_k's candidate set, and multiplying those terms into the likelihood $P(\{X_k\})$. Thus, the Bayes factor has the same decoupled structure as the likelihood, which allows us to find the assembly that maximizes it efficiently, using Dynamic Programming.

As in the case of the single tree, the main idea is that, conditional on a particular primitive selected as X_k in an assembly, the likelihood is a product of two parts – one involving only the nodes of the subtree rooted at X_k, and the other involving only the other nodes – and each of the two can be maximized independently. Therefore, we perform optimization using a Viterbi algorithm on tree T, visiting children before parents. The optimization at a node involves the selection of not only the best primitives to choose from the children's candidate sets, but also of the edges to be included in the tree, since a node can be a child of X_k in a mixture component whenever it is a *descendant* of X_k in the generating tree T.

The total running time of this optimization is, as in the single-tree case, $O(M^2\#\text{edges})$, where M is the size of candidate sets, and $\#\text{edges} = O(K^2)$ since we need to consider all the edges joining a node of T with its descendant. Maximization of the Bayes factor allows us not only to localize an object if it's present, but also to perform detection: To determine if the image contains the object of interest, we compare the maximum value of the Bayes factor with a threshold.

7. Tree root as a coordinate frame

In the above discussion, we required that the root X_{root} always be a part of the object, so that each mixture component has a connected structure. However, we would prefer to allow *any* primitive to be absent. We achieve this by introducing a new primitive type, X_0, and make it the root of the tree. This primitive is not detected in the image; rather, it can be thought of as a *global coordinate frame* roughly representing the configuration of the object as a whole (such as the position and orientation of a person in the image). We add X_0 to each training assembly during learning; for detection, a number of candidates are created that span the necessary range of global configurations (in case of detecting people, this means a range of positions and orientation, rather widely spaced because X_0 represents the global configuration only loosely in the training data).

In addition to keeping the mixture trees connected, the inclusion of the global frame allows us to specify distributions invariant with respect to transformations such as rotations and translations. For example, the likelihood of a body configuration should not change if the entire configuration, including X_0, is translated and rotated. This invariance is achieved by associating a *local coordinate frame* with each primitive X_j, and making the conditional $P(X_k \mid X_j)$ functions of the *relative configuration* of X_k in X_j's coordinate frame. Because X_0 is the root of the tree, the configuration of the entire object is relative to X_0, and we make our distribution invariant to a set of transformations by conditioning it on the global frame of reference, that is, replacing $P(\{X_k : x \in S\})$ with $P(\{X_k : x \in S\} \mid X_0)$.

8. Tracking

We have applied our model of a person to human figure tracking. Tracking is a difficult problem, as both spatial and temporal constraints must be combined: we need to find objects that both look like people and move like people. Often, the problem is made easier by making the model tighter in the spatial dimension (manual tracker initialization) and in the temporal one (tight motion model). While these simplifications make tracking easier, they seem somewhat unreasonable for many applications: if a user wants to find moving people in video sequences, he should not have to manually mark their joint locations or specify the precise way in which they move. In fact, the data we are using, collected by Muybridge [5] over 100 years ago, has a very low and variable frame rate and a rather loose alignment between frames (which are photographs taken by separate cameras, synchronized to go off in a sequence), which makes a tight motion model inapplicable. The large limb motions between frames also mean that the pose in a frame specifies only a vague prior for the next frame, and thus the resampling step of particle filtering [6] may involve an expensive search. On the other extreme, one can model only the spatial configuration of a person and ignore temporal links (e.g., [7], who perform tracking independently for pairs of frames, using local motion fields). This approach, while efficient, is unable to enforce motion consistency over time. For example, a human body can be found in the left half of a frame and the right have of the following frame, even though such motion is not physically possible.

We deal with these problems by combining a mixture-of-trees model for a person with a weak motion model, which does not assume any particular type of activity, but simply bounds the motion of each body part. While exact inference in this model is difficult, it can be done approximately, by alternating optimization over space and over time.

8.1. Detecting body parts

People are modeled as assemblies of body parts. We consider 9 types of body parts: the torso, and the upper and lower halves of each limb. We assume to know the scale of the person and thus can look for body parts as roughly rectangular image segments of a fixed size that satisfy certain appearance constraints. The models photographed by Muybridge wear little or no clothes, thus allowing us to ignore the problems associated with loose or textured clothing. We

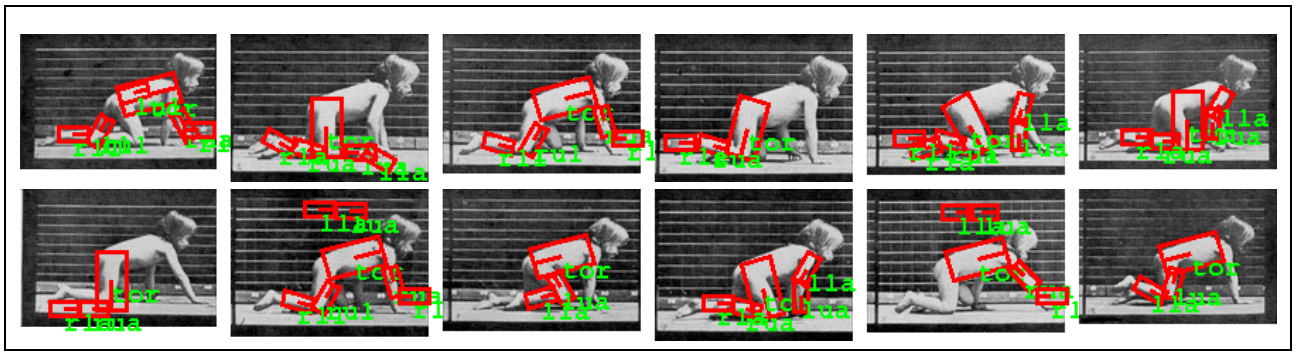


Figure 2: *Motion information is essential to inferring the person's configuration correctly. Temporal coherence was not used for tracking this sequence, thus the best assembly was independently found for each frame. The big red rectangle represents the torso, while the smaller ones correspond to limbs. The resulting track is not physically possible, as illustrated by the motion of the torso.*

detect limbs by template matching, using two separate templates for the torso and the limbs, each of which emphasizes long regions that are bright along the axis and darker closer to the boundaries and have edges of an appropriate orientation on either side. The body part locations are found by convolving the image with a template in a range of orientations, followed by non-maximum suppression of responses. We keep the body parts where a local maximum with respect to position and orientation is achieved, provided the response value is sufficiently large.

Since each body part has a direction (so that we know which end is the shoulder and which is the elbow), each of the detected image regions is converted into 2 candidate body parts, with the opposite orientations. Each primitive is parameterized by the coordinates of its two ends. Such a parameterization allows us to define simple conditionals, even though it is redundant (since the size of each limb is fixed).

8.2. Modeling the body

In order to learn a mixture-of-trees model for a human body, we must specify the form of conditionals $P(X_k \mid X_j)$. We want our model to be invariant with respect to translation and rotation. This is achieved if the conditionals are represented as $P(X_k \mid X_j) = P(u_1, v_1, u_2, v_2)$ where (u_i, v_i) are the coordinates of the ith end of the segment represented by X_k, in the frame of reference positioned and oriented according to the segment X_j. This is in fact a conditional distribution for X_k since (u_i, v_i) are obtained from the parameters of X_k by rotation and translation, which preserve volume. The distribution $P(u_1, v_1, u_2, v_2)$ is modeled as a Gaussian with a full covariance matrix, and is learned from a set of training assemblies, obtained by detecting candidate segments in a set of images, and manually selecting the correct limbs. The training data set was quite different from the test set. First, the images used for training were taken from a book of models [9], featuring people wearing swimsuits against white background. Only frontal views of standing people were used for training (unlike the Muybridge sequences used in testing, which contain lateral views as well), and the body parts used for training were detected using a different method than described above. Nevertheless, we have obtained a model of a human body that allows a wide range of configurations and captures connectivity relations among human limbs. A mixture of trees was learned by minimizing entropy; not surprisingly, the root X_0 has the torso as the only child, whose children were the upper arms and legs, each of which has the corresponding lower limb as the child. This tree structure is widely used (e.g., [2]) but now also shown to be optimal in terms of the amount of information it captures.

8.3. Modeling a moving person

Consider an image sequence containing a moving person. We will represent the person in each of the F frames as an assembly of primitives $A_1 \dots A_F$, and define the likelihood of the sequence to be

$$P(A_1 \dots A_F \mid \text{track}) \propto \prod_{f=1}^{F} P(A_f) \times P_{\text{temp}}(A_1 \dots A_F),$$

where $P(A_f) = P(A_f \mid \text{person}, X_0)$ is the likelihood of an individual assembly and $P_{\text{temp}}(A_1 \dots A_F)$ is the term that ensures temporal coherence. We use an indicator function such that $P_{\text{temp}}(\cdot) = 1$ if the motion of each limb between any two frames does not exceed a certain bound (which grows with the time interval between the frames), and $= 0$ if these constraints are not satisfied.

The temporal constraints captured in $P_{\text{temp}}(\cdot)$ are essential for enforcing motion coherence. Although ignoring them (e.g. [7]) would result in a model that allows efficient exact inference, tracking in difficult sequences

In our experiments, we consider the problem of extracting the assembly sequence most likely to correspond to a moving person. This involves maximization of the Bayes factor $B(A_1 \dots A_F) =$

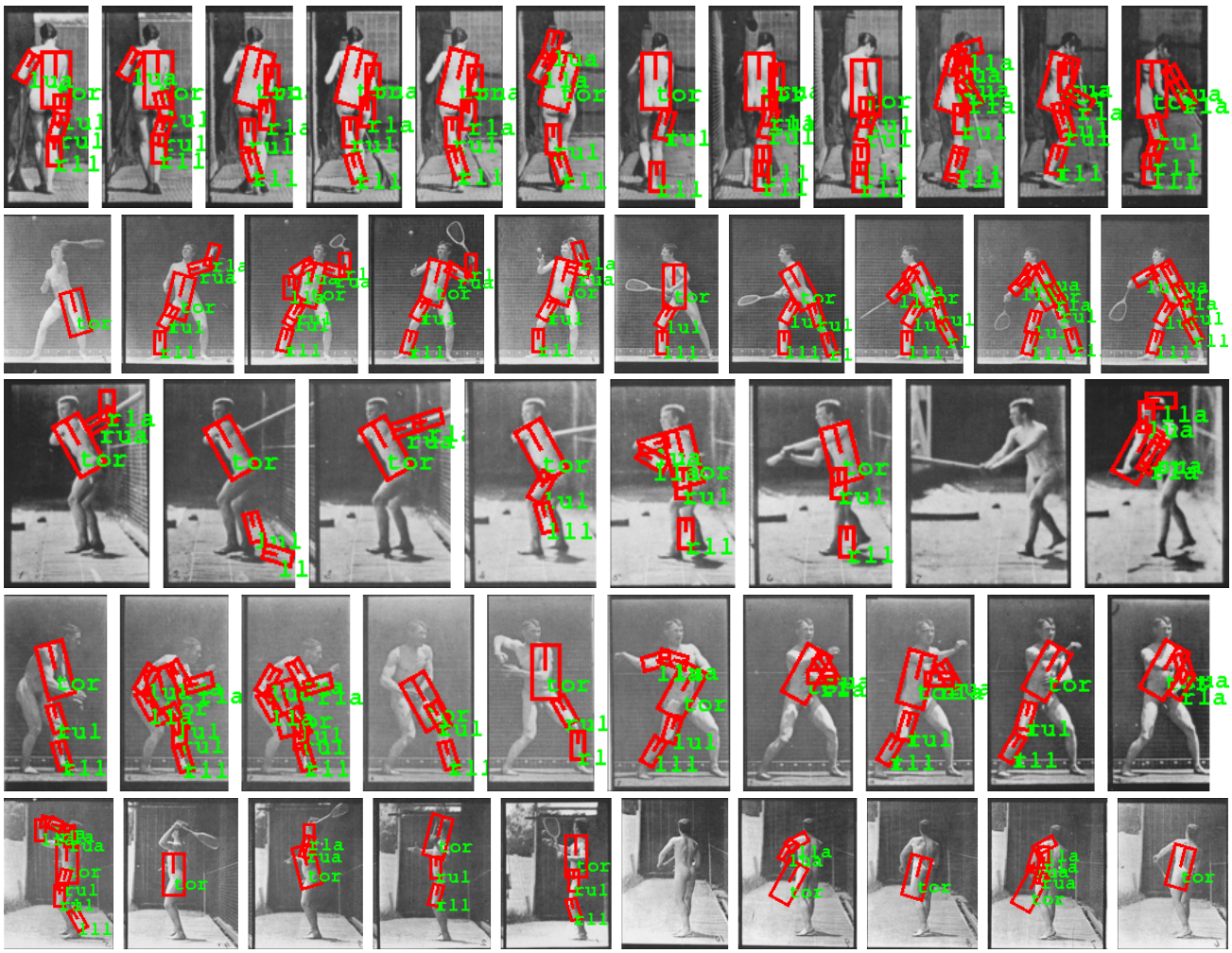


Figure 3: *Examples of tracking a person. In all of the sequences the tracker has correctly determined the general position of the person, even though they engage in a variety of activities and undergo changes in aspect. Top two sequences are correctly tracked, while the others also contain some frames where the inferred configuration is incorrect.*

$P(A_1 \ldots A_F \mid \text{track})/P_{neg}(A_1 \ldots A_F)$ where P_{neg} is the distribution for an assembly sequence not corresponding to a person. As in the single-image case, we model segment configurations in non-human assemblies as independent; therefore, the Bayes factor decouples:

$$B(A_1 \ldots A_F) \propto \prod_{f=1}^{F} \frac{P(A_f)}{P_{neg}(A_f)} \times P_{\text{temp}}(A_1 \ldots A_F).$$

In the absence of temporal constraints, this would be simply the product of single-frame Bayes factors $B(A_f) = P(A_f)/P_{neg}(A_f)$ which do not interact with each other and could thus be maximized independently as in Section 6. However, ignoring temporal constraints results in sequences of assemblies with an inconsistent temporal behavior, which could not correspond to a moving person (figure 2).

The constraints on limb motions add links to our model that make maximization difficult: each limb is now connected not only to all of its ancestors and descendants in its own frame, but also to the limbs of the same type in other frames. Thus, all of those will interact, resulting in large groups of mutually interacting variables (cliques) and making exact inference in such a model very expensive. Nevertheless, we can obtain a *local maximum* of the Bayes factor by performing a sort of a *coordinate ascent*. Each ascent step maximizes the Bayes factor with respect to some of the limbs while keeping the rest unchanged. We use two types of an ascent step: a *spatial* step, where we maximize the Bayes factor with respect to the configuration A_f in frame f, while keeping assemblies in the other frames unchanged; and a *temporal* step, where we choose the best configuration for a body part of a particular type in each frame, without

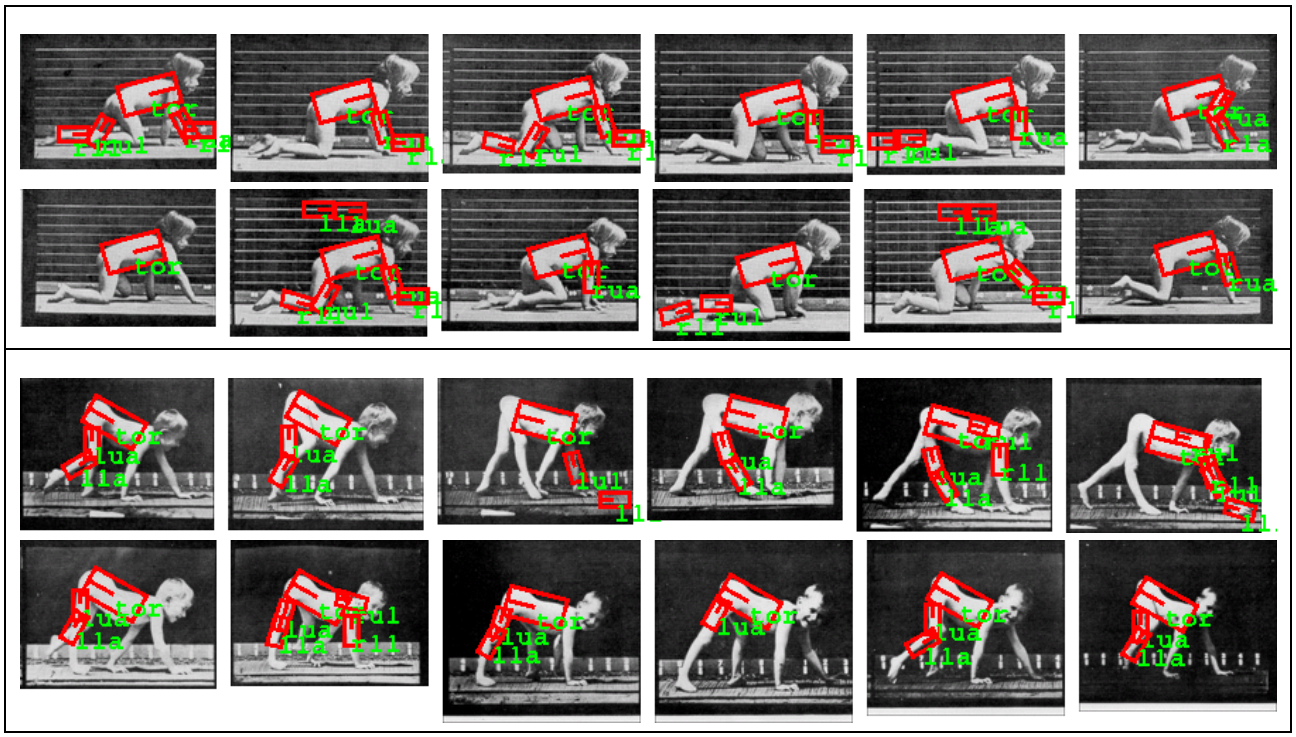


Figure 4: *Tracking crawling children. In both sequences, the tracker attempted to automatically infer the configuration of a person. Even though the training data contained neither children nor crawling people, the track has been identified correctly in the top sequence. In the bottom sequence, the tracker confused the arms and the legs, which would not happen if the head was used as one of the primitives. Comparing the top sequence with the one in figure 2 clearly demonstrates the importance of temporal constraints, imposed by the loose motion model, in tracking the configuration correctly.*

disturbing the other body parts. Both of these steps can be made efficiently with a Viterbi algorithm.

Maximizing the Bayes factor over space: Given assemblies in each frame except A_f, it is easy to choose the configuration A_f that maximizes $B(A_1 \ldots A_F)$. First, a set of candidate body parts of each type in frame f is narrowed by eliminating those inconsistent with the assemblies in other frames. Then, the single-image Bayes factor $B(A_f)$ is maximized, as in Section 6, by selecting body parts from the narrowed candidate sets.

Maximizing the Bayes factor over time: Let us use Torso as an example of the body part whose configuration is to be optimized in all the frames. Given a set of candidate parts of each type in each frame, we look for a sequence of assemblies such that the product of the single-frame Bayes factors, $\prod_{f=1}^{F} B(A_f) = \prod_{f=1}^{F} P(A_f)/P_{neg}(A_f)$, is maximized, subject to temporal coherence of the Torso. As the first stage of this optimization, we maximize $B(A_f)$ separately for each frame and each choice of Torso (including its absence), using a version of the method of Section 6. Then, we choose a Torso (possibly a missing one) in each frame so that the product of corresponding Bayes factors is maximized, subject to the constraints on motion of the Torso. This step is accomplished with a Viterbi algorithm.

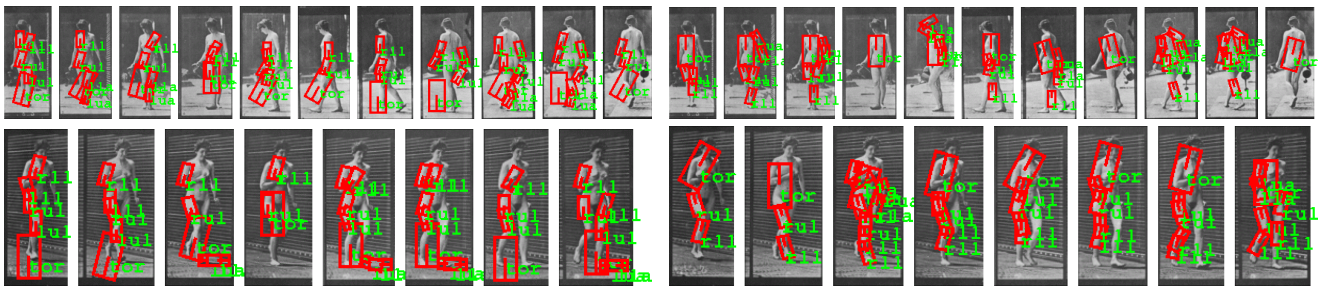
To maximize the Bayes factor $B(A_1 \ldots A_F)$, we first choose an initial sequence of assemblies, by considering body parts in order and choosing the best configuration for that part in each of the frames, subject to the temporal constraints. Then, the Bayes factor is repeatedly maximized over space and time until convergence to a local maximum. In our experiments, we augmented the model and the optimization procedure to allow an assembly to be missing in an image. We associate a likelihood $P(\emptyset)$ with a missing assembly, which limits the damage that a poorly detected assembly can do to the likelihood of the whole sequence.

While our procedure maximizes a function, a few simple changes (such as replacing maximization with summation) convert it into a Markov Chain Monte-Carlo sampler. By drawing samples of the assembly sequence, we would be able to preserve uncertainty so that we could incorporate new information, such as more accurate motion models, into our results later.

8.4. Results

We have applied the mixture-of-trees model combined with a weak motion model to track a variety of sequences from the Muybridge collection.

Figure 3 shows several sequences with the tracked as-



a b

Figure 5: *The global frame of reference can be used to impose constraints on the configuration. In* (a)*, the tracker has failed if the orientation of the body is not constrained. By making all of the candidate global frames (i.e., the root X_0 of the tree) upright, the tracks in* (b) *are obtained, which represent the person better.*

semblies overlaid on top. For these sequences, the candidate global frames of reference X_0 (not shown) were placed at a grid of positions, in the 4 directions spaced by 90°, which effectively makes recognition translation- and scale-invariant, since the global frame only loosely represents the configuration of the body in the training data. People are tracked in a variety of activities; in all of the shown sequences the body position is correctly tracked, although in some frame the configuration is not correctly represented.

The global frame of reference can be used to impose constraints on the rough location and orientation of a person. For example, if we know that the person we track is upright, only the upright global frame needs to be used. This is illustrated in figure 5. Figure 5(a) shows several sequences where the tracker has failed if all 4 directions for the global frame were used. All of those tracks can be corrected by requiring that the global frame be upright, and the results are shown in figure 5(b).

Because of the weak configuration and motion model, our tracker is able to follow motions it has not seen before. In figure 4, two sequences of crawling children (photographed by Muybridge [4] over a century ago) are tracked. The healthy child is tracked correctly, even though there were no children or crawling subjects in the training data. The handicapped child is tracked incorrectly, since his arms and legs are confused by the tracker – but, up to that flip, the configuration is correct. In both of the sequences, we had to reject the candidate torsos found in the lower third of the image, due to the spurious torsos and limbs found in the floor tiles and giving rise to human assemblies.

Comparing figure 4(top) with figure 2 illustrates the importance of temporal coherence in tracking the body correctly. These two figures contain the same sequence, but the temporal constraints were ignored to obtain the track in figure 2. Thus, the performance of the tracker deteriorates dramatically if a motion model, even a weak one, is not used.

References

[1] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 8–15, 1998.

[2] P. Felzenschwab and D. Huttenlocher. Efficient matching of pictorial structures. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2000.

[3] M. Meila and M.I. Jordan. Learning with mixtures of trees. 2000. In review, Journal of Machine Learning Research.

[4] Eadweard Muybridge. *Animals in Motion*. Dover, 1957.

[5] Eadweard Muybridge. *The Human Figure in Motion*. Dover, 1989.

[6] Hedvig Sidenbladh, Michael J. Black, and David J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *European Conference on Computer Vision*, 2000.

[7] Yang Song, Xiaolin Feng, and P. Perona. Towards detection of human motion. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 810–17, 2000.

[8] R.E. Tarjan. Finding optimum branchings. *Networks*, 7(1):25–36, 1977.

[9] unknown. *Pose file*, volume 1-7. Books Nippan, 1993-1996. A collection of photographs of human models, annotated in Japanese.

[10] M. Weber, W. Einhauser, M. Welling, and P. Perona. Viewpoint-invariant learning and detection of human heads. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 20–7, 2000.