Augmenting Physical State Prediction through Structured Activity Inference

Nam N. Vo and Aaron F. Bobick

***Abstract*— We address the problem of predicting the physical state of a an agent performing a known activity. In particular we are interested in predicting human movement during complex composite activities. Our proposed framework combines a graphical model that extends the Sequential Interval Network (SIN) [1] for modeling global temporal structure of activities with a low level dynamic system for modeling the dynamics of the physical state. Specifically, two sets of new hidden state variables are added: one with respect to the temporal structure and one with respect to time. A mapping factor is defined to ensure these variables values remain consistent and hence allows fusing the two sources of information. We then derive an inference algorithm for computing the posterior densities of the hidden variables. The system can run in an on-line predictive mode to recognize on-going activity and make predictions arbitrarily far in the future during execution of the activity. Experiments illustrate that the long term prediction performance benefits from the knowledge about the temporal structure of the activity while short term prediction performance is improved by incorporating the dynamics of physical state.**

I. Introduction

Being able to recognize and understand human activity as it is occurring, as well as predicting future actions is essential for safe and collaborative interaction between humans and robots. With such abilities, a robot can make long term plans to assist a human at the appropriate time as well as decrease the likelihood of interfering with the human activity. Recognizing and predicting complex process of human activity is also fundamental in other domains such as surveillance, work-flow monitoring, anomaly detection, and skill assessment. This task is challenging in real-world scenarios as there are typically variations in the execution of the activity and ambiguity in sensing both the human and the environment.

Here we consider the problem of parsing composite human activities that follow specified patterns. An activity is defined as a partially ordered, sequence of both required and optional primitive actions. One common task explored in vision research is that of frame labeling i.e. infer the discrete hidden state of what action is occurring at each time-step. In such systems, various uncertain physical measurements are leveraged as being indicative of the true, but hidden, underlying physical state. Examples might include instantaneous skeleton estimation or even simply location. Therefore, one can consider a second task: infer these hidden physical states during the activity and, more importantly, predict their state in the future.

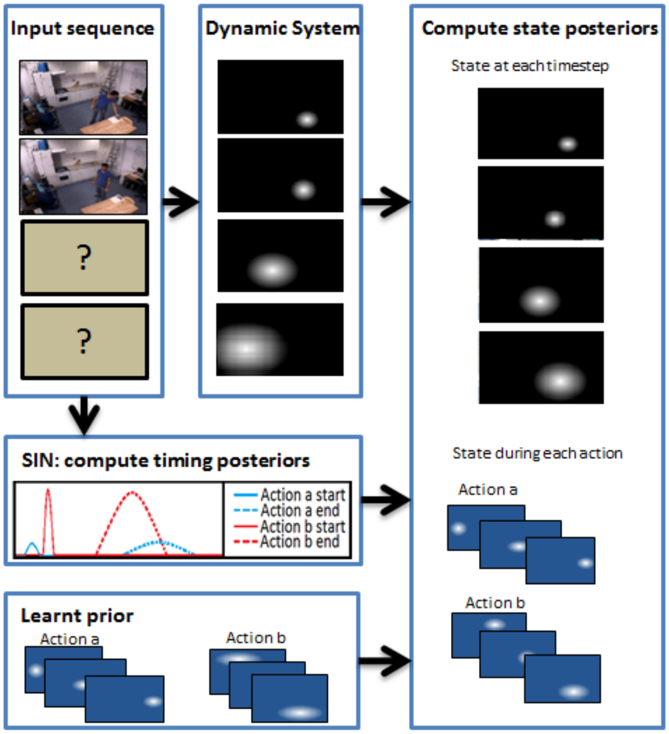


Fig. 1. The pipeline of our system. The input is a sequence that might be partially observed. The outputs are the posterior of the timings (when each action starts or ends) and the posteriors of the physical state at each time-step or during each action.

We will assume an activity has a known, global temporal structure — we will define the structure shortly — and that each action within the activity imposes a prior distribution on the physical state (e.g. the action [get-the-spoon] involves the person goes to the kitchen cabinet's position). Our goal is to incorporate this knowledge of the temporal structure to improve the accuracy of predictions based solely on physical properties such as velocity or simply priors based upon current physical state such as location.

Our proposed framework is built upon the Sequential Interval Network (SIN) [1], an alternative to HMMs or CRFs that can model the global temporal structure of an activity, perform reasoning on interval/segment level and still permit exact inference. The key to that work is that the variables of inference are the start and end times of each action component. In the work here, we augment the network to model physical state with hidden continuous variables. At the lower, evidence level, a dynamic system,

The authors are with the Georgia Institute of Technology, Atlanta, GA contactable at namvo@gatech.edu, afb@cc.gatech.edu

as part of the observation model, is employed to estimate the physical state at each time-step. We learn the prior distribution of the state during each action, then proceed to perform inference to obtain the posterior densities of the physical state, with respect to time and each action. The pipeline of the system is shown in figure 1. The basic idea is this: physical measurements provide strong constraint on near term physical state whereas the activity structure provides weaker but longer term information. Our inference model automatically incorporates both types of knowledge for making future physical state predictions.

We organize this paper as follows. The related work is discussed in the next section. After reviewing the SIN model in section III, we introduce our extension of the graphical model in section IV. In section V and VI, we perform different activity recognition experiments with our method using both publicly available datasets and our own assembly task data. Finally we conclude in section VII.

II. Related Works

There has been a growing number of works on recognizing and predicting structured events in computer vision literature. In [2], [3], an AND-OR grammar is used to learn and parse activity sequences. Similarly, Li et al [4] used Probabilistic Suffix Tree to do early detection. In [5], [6], inverse optimal control is used to predict human movement based on environment's feature. Here we are interested in both predicting the action label and the continuous dynamic state such as human pose or movement during the activity.

In robotics, recognition and prediction play a critical role for fluent human-robot interactions. Huber et al [7] demonstrated such interaction where the robot was shown to be more helpful given complete knowledge about the task and sensing. In [8], [9], a cost based framework is proposed that makes decisions based on anticipation and which improves task efficiency compared to purely reactive process. However these systems are not specifically designed to handle perceptual ambiguity when recognizing and predicting the activity process.

A significant number of methods for modelling random process are time sliced graphical model. Classical examples are Linear Dynamic System (LDS) and Hidden Markov Model (HMM), both of which can be represented as a Dynamic Bayes Network (DBN). They have been used for object tracking [10] or action recognition [11], [12].

More sophisticated systems are needed to model complex sequences. Hidden semi-Markov Model (HSMM) and segmental model [13], [14] are preferred to avoid the limiting geometric duration assumption of HMM. It has been used in [15] for complex action detection. Switching LDS and segmental switching LDS (SSLDS) were introduced to model complex nonlinear patterns as a mixture of linear ones. One can think of the switching variable in these models as HMM state. SSLDS have been used for inferring or synthesizing motion pattern [16], [17], [18]. However exact inference on these graphical models is difficult to achieve.

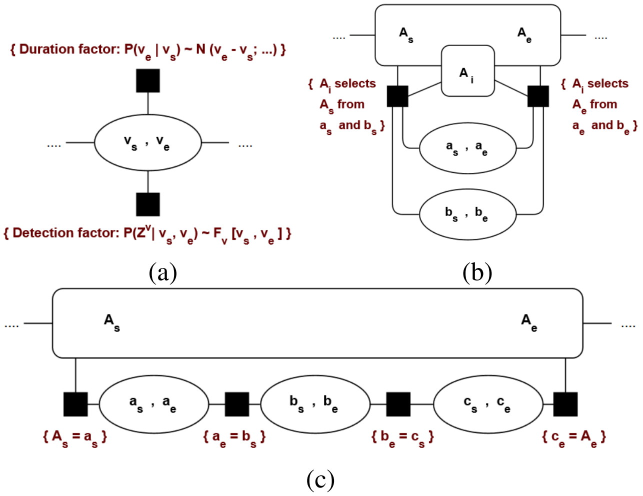

{ Duration factor: P(v_e | v_s) ~ N (v_e - v_s; ...) }
.... v_s , v_e ....
{ Detection factor: P(Z^v| v_s, v_e) ~ F_v [v_s , v_e] }
(a)
.... A_s A_e ....
A_i
{ A_i selects A_s from a_s and b_s }
{ A_i selects A_e from a_e and b_e }
a_s , a_e
b_s , b_e
(b)
.... A_s A_e ....
a_s , a_e
b_s , b_e
c_s , c_e
{ A_s = a_s }
{ a_e = b_s }
{ b_e = c_s }
{ c_e = A_e }
(c)

Fig. 2. The SIN model construction in factor graph representation: (a) the primitive [v], (b) the OR-composition [A is a or b], and (c) the AND-composition [A is a and b and c]. Note that the sub-components of the composition can be either primitive or compositions themselves. In addition, all factors in the compositions are just deterministic constraints.

Other DBN and variant approaches for parsing human activity are [19], [20], [21], [22]. With complex state space, these approaches have to use different sampling techniques to perform inference. In [1], we introduced SIN to alleviate this problem for long activity sequence whose composition can be represented as an iterated structuring of AND-OR elements. Because we require a finite length activity, the representation is not formally a context free or even regular grammar, but for modeling finite activities it resembles a stochastic CFG. By taking advantage of this knowledge about the temporal structure, modeling the sequence in time sliced manner can be avoided. More importantly, prediction then can be made for any time-step including ones in arbitrary far future. This model is described in the next section.

III. The Sequential Interval Network

In this section, we briefly review the SIN model. For more detail, we refer the readers to [1] (source code is available). An earlier version of this model was applied in a human-robot collaboration framework [23], [24].

SIN is a graphical model for parsing time-series data. However unlike common temporal graphical models, a first order Markov property is not assumed, hence its capability is similar to that of a segmental model [13], [14] while allowing exact inference.

A. Represent the temporal composition of the activity by a finite grammar

Our graphical model representation of an activity is generated from a probabilistic, finite grammar that specifies the temporal structure. We assume the activity is composed of sub-activities or actions, which can in turn be compositions themselves. Hence each activity/action can be either a composition, which correspond to non-terminals in the grammar, or primitive, which corresponds to terminals. The grammar's production rules, including the AND-rule and the OR-rule, define the compositions.

Different from traditional time sliced graphical models (HMM, DBN, temporal CRF), the generated network models the timings of the actions instead of the state of each time-step; thus allowing reasoning at interval or segment level (for example: duration). Specifically, for each action A (which can be either *primitive* or *composition*), two hidden variables are defined: A_s is the time-step when A starts and A_e is the time-step when A ends, where these variables obtain discrete values between 1 and T, the maximum length of the activity, or the special value (-1), which means action A does not happen. Then we can define factors such as $P(A_e|A_s)$ to model the duration and $P(Z^A|A_s, A_e)$ to model the observation of the action if A is a primitive (figure 2 (a)).

In the case where A is a composition, it can be either the AND-composition or the OR-composition according to the grammar's production rule. The AND-composition defines A to be a sequence of sub-actions. The sub-actions are constructed recursively, then dependencies are added to the network to express the temporal constraints: the end of a action is the start of the next one, the start and end of A are the start of the first and the end of the last action in the sub-action sequence respectively (figure 2 (c)).

The OR-composition defines action A to be one of its sub-actions (which means the other sub-actions do not happen). This important composition allows for variation in the activity temporal structure. Similar to the AND-composition, the network for sub-actions is constructed, then a selector variable A_i is added with appropriate multiplexer condition probability tables to constrain that the start and the end of A must have the same value as one of its sub-action, while others have the special -1 value (figure 2 (b)).

B. Inference

From the grammar, the network is generated in a recursive manner. Now all the conditional probability tables (except for deterministic constraints) must be computed: $P(A_e|A_s)$ and $P(Z^A|A_s, A_e)$ for every primitive A. For the first factor, we will use Gaussian distribution to model the duration of primitive action and learned parameters from labeled training example. The second factor requires having visual detectors that are applied to the the input sequence and which returns the likelihood of every interval. As will be described below, these detectors do not need to be very discriminating and can be easily learned from limited training data. Given these elements, inference can be performed.

Notice that the network is actually composed of multiple chain-like structures put together by deterministic constraints (AND-composition and OR-composition). Hence a forward-backward message passing (belief propagation) algorithm is proposed to do exact inference. The posteriors of the hidden variables ($P(A_s, A_e|Z)$ for every action A) are then obtained. This tell us the likelihood that action A happens and if so a density over when.

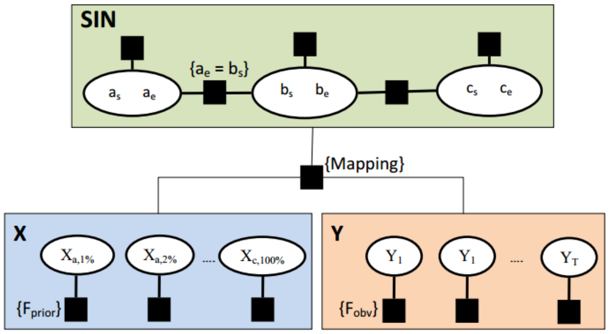


Fig. 3. Our proposed graphical model in factor graph representation. In this example, the activity is assumed to be a sequence of 3 primitive actions: a, b and c. SIN is on the top, modeling the timing of the actions. Two sets of continuous variables are at the bottom modeling physical state. The factor {mapping} keeps these variables consistent with each other.

SIN models the entire activity and are most naturally thought of as an offline process, i.e. the entire input sequence is provided and inference is performed. However, and most important for the prediction work developed here, it can be adapted to run in an on-line, streaming mode. To accomplish this, at any moment in time, we pretend the whole sequence is available, and fill in the missing future entries in the factor tables with a default expected value. When more observations become available, the primitive action detectors are run again. Hence these entries' values can be recomputed and the inference can be performed again. Notice that this makes the inference at each time-step independent of each other.

In [1], we demonstrated how to perform recognition, prediction and temporally segment an input sequence into component actions.

IV. Prediction of Physical State

In this section, we extend our method to include the estimation or prediction of some physical states. In the context of human activity recognition, this state could be body positions, velocity or pose. At any moment in time, the physical state is tied to the primitive action currently being performed. This approach is similar to ideas in the SSLDS work mentioned earlier. For example, the action "getting the spoon" will involve the human moving from the table to the cabinet. Therefore after predicting when such action is going to happen using SIN model, prediction about future human position can also be made.

A. Network variables

Most existing approaches represent such physical states by modeling the state at each time-step. Differently, we model the physical state by two sets of new hidden variables: X representing the physical state with respect to primitive action completion stage (defined below), and Y representing the physical with respect to time, as shown in figure 3. Since there is ambiguity as to which action is happening at which time-step, there will be a probabilistic mapping between these two sets of random variables. The goal is to infer their posteriors given available observations. And recall the posteriors are for all time regardless of the current time up to which real observations have been made. This is how the system predicts future state.

First, we introduce $X = \{X_{\alpha,c}\}$ for every primitive action α in the activity and every completion stage c belonging to a predefined completion stage set { 1%, 2%, ... 100%}, where $X_{\alpha,c}$ represents the physical state at the time-step corresponding to completion stage c during the action α. For example $X_{\alpha,1\%}$ and $X_{\alpha,100\%}$ are the physical state when action α starts and when action α ends respectively. Different instances of the same action could have different timings and durations, this representation offer invariance to those properties. More over, we assume that each primitive action imposes a prior distribution on the physical state during that action, hence even for different instances of the action, the physical state at the same completion stage would be similar, so the representation enables learning this property.

Next, similar to typical time sliced graphical model, we also define the variable Y as: $Y = \{Y_t\}$ for every time-step t between 1 and T, where T is the maximum length of the activity and Y_t will be the physical state at time-step t.

B. *Network factors*

Before introducing the inference, we describe the factors/potential functions involving X and Y, shown in figure 3.

First, we can learn the X distribution from the training data assuming both action timing and physical state annotation are provided. This is presented as factor F_{prior} in the graph. To make the computation efficient, we will assume the variables $X_{\alpha,c}$ are normal distributed and conditionally independent (given the available observation and other hidden variables). Thus the factor for each variable $X_{\alpha,c}$ is: $F_{prior}(X_{\alpha,c}) = N(X_{\alpha,c}; \mu_{\alpha,c}, \sigma_{\alpha,c})$, where $\mu_{\alpha,c}$ and $\sigma_{\alpha,c}$ are parameters computed during training.

Similarly, we assume Y_t are normally distributed and conditionally independent. This is presented as the factor: $F_{obv}(Y_t) = N(Y_t; \hat{\mu_t}, \hat{\sigma_t})$, where $\hat{\mu_t}, \hat{\sigma_t}$ is the estimate at of time-step t based only on the available physcial obervations. This can be done by applying any method that models the low level dynamics of the system and takes into account the observation. In our experiments, we choose to use the simple Kalman smoothing on top of the raw measurements. This processing step allows incorporation of all measurements under a physical model (in our cases, we will use the constant velocity model) to output a probabilistic estimation $N(\hat{\mu_t}, \hat{\sigma_t})$ for every time-step t. Hence factor F_{obv} indirectly takes into account all this information.

Finally, the special deterministic factor {Mapping} resolves the ambiguity between the time and the completion stage. It assumes a linear scale relationship between two of them (which is plausible since the actions are defined to be ”primitive”, hence assumed to progress at a constant rate). Practically, the factor maps $X_{\alpha,c}$ to the corresponding Y_t given the timing α_s, α_e of action α (which means $t = \alpha_s + (\alpha_e - \alpha_s).c$), and vice versa. For example if $\alpha_s = 100$ and $\alpha_e = 300$, then Y_{100}, Y_{200} and Y_{300} must have the same values as $X_{\alpha,1\%}$, $X_{\alpha,50\%}$ and $X_{\alpha,100\%}$ respectively. This {Mapping} factor allows message passing between these 2 sets of variables: a factor F_{prior} on variable $X_{\alpha,1\%}$ can be used as a prior distribution on variable Y_{100}.

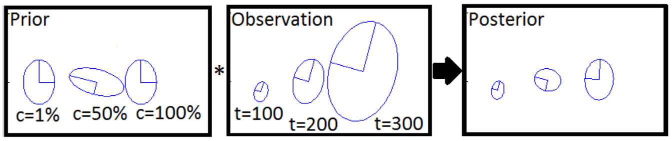


Fig. 4. In this example, the physical state is 2D position. Assume $\alpha_s = 100$ and $\alpha_e = 300$, then $X_{\alpha,1\%}$, $X_{\alpha,50\%}$ and $X_{\alpha,100\%}$ have the same values as Y_{100}, Y_{200} and Y_{300} respectively. (a) shows 3 ellipsoids representing the 3 Gaussian distributions F_{prior} factors of $X_{\alpha,1\%}$, $X_{\alpha,50\%}$ and $X_{\alpha,100\%}$, (b) shows 3 F_{obv} factors of Y_{100}, Y_{200} and Y_{300} (c) By multiplying corresponding pairs of factors, we obtain the posteriors: $X_{\alpha,1\%}$ and Y_{100}, $X_{\alpha,50\%}$ and Y_{200}, $X_{\alpha,100\%}$ and Y_{300}.

C. *Inference*

Algorithm 1 Physical State Inference

```
Input: $P(\alpha_s, \alpha_e|Z)$, $(\mu_{\alpha,c}, \sigma_{\alpha,c})$ and $(\hat{\mu_t}, \hat{\sigma_t})$ for every primitive $\alpha$, its completion stage c and time-step t.
Initialize $x_{\alpha,c}$ and $y_t$ as empty sets (for every $\alpha$, c, t)
For each primitive action $\alpha$:
    For every value of $(\alpha_s, \alpha_e)$:
        For each completion stage c:
            Given $\alpha_s$, $\alpha_e$, $c$, compute the corresponding time-step t.
            Multiply $N(\mu_{\alpha,c}, \sigma_{\alpha,c})$ and $N(\hat{\mu_t}, \hat{\sigma_t})$ to form the scaled Gaussian $N(\mu, \sigma)$ with scale factor s.
            Add the result Gaussian to $x_{\alpha,c}$ with the weight $P(\alpha_s, \alpha_e|Z)$.
        For each time-step t between $\alpha_s$ and $\alpha_e$:
            Given $\alpha_s$, $\alpha_e$, $t$, compute the corresponding completion stage c (if c is not in the set, interpolate $N(\mu_{\alpha,c}, \sigma_{\alpha,c})$).
            Multiply $N(\mu_{\alpha,c}, \sigma_{\alpha,c})$ and $N(\hat{\mu_t}, \hat{\sigma_t})$ to form the scaled Gaussian $N(\mu, \sigma)$ with scale factor s.
            Add the result Gaussian to $y_t$ with the weight $P(\alpha_s, \alpha_e|Z)$.
Output: $x_{\alpha,c}$ and $y_t$ are posteriors of $X_{\alpha,c}$ and $Y_t$. (the distributions are in form of mixture of Gaussians)
```

Consider the simple scenario where the timings are known, hence the mapping between X and Y is resolved. For each pair of corresponding $X_{\alpha,c}$ and Y_t, one can interpret them as a single variable with the prior distribution F_{prior} and the likelihood F_{obv}. In this case both hidden variables will have the same posterior density which is the normalized product of the prior and the likelihood. In our implementation, this density will be a Gaussian distribution since both F_{prior} and F_{obv} are Gaussians. An illustrated example is shown in figure 4.

In general, the precise timings are not known. Instead, we compute their posteriors with SIN ($P(A_s, A_e|Z)$ for every action A). In this case, inference involves integral of all possible timings (exact inference) or the likely ones (sampling inference). The exact inference is shown in Algorithm 1. The output is the posterior densities of $X_{\alpha,c}$ and Y_t as mixture of Gaussians, instead of a single Gaussian like the last case. This result can be difficult to use since the mixture is big, so in our experiments, instead of saving the mixture's

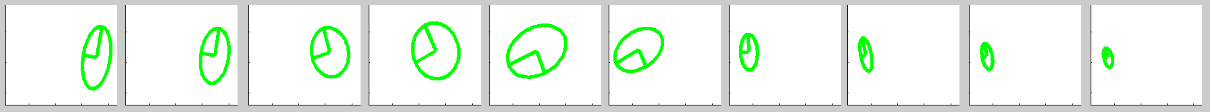
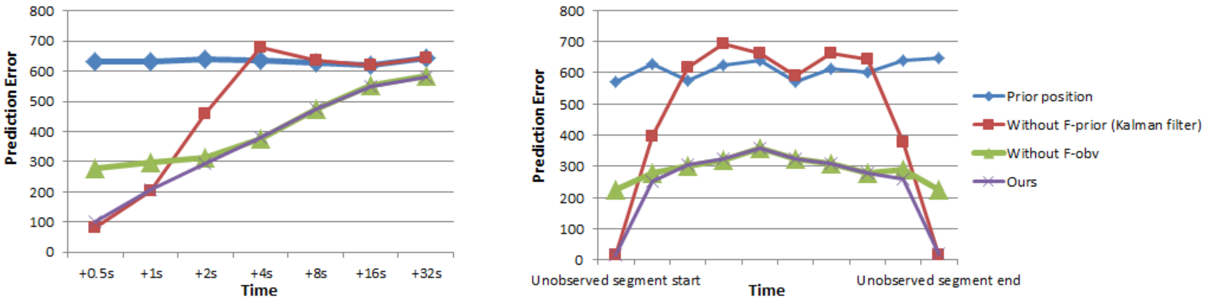


Fig. 5. Learnt distribution of human position at 10 completion stage (1%, 10%, 20%, ... 100%) of action get-the-spoon. In this action, the human operator would move from the table (on the right) to the kitchen drawer (on the left) and get the spoon inside.



(a)

(b)

Fig. 6. TUM kitchen dataset experiment results: (a) Future human position prediction: Early future predictions — less than one second — are dominated by the Kalman observations (red squares) whereas later predictions are controlled by the action predictions (green triangles). Our method naturally combines the two. The Kalman only estimate error was bounded by a simple heuristic to prevent it from expanding beyond the domain. (b) Human position smoothing shows a similar pattern.

parameters, we convert it into a probability mass function in form of a heatmap grid by sampling.

Note that in the inference algorithm, we keep the calculation of the posteriors of X and Y separate. This makes it efficient in case only one of them is desired, the calculation of the other can be skipped. In addition, it is not mandatory to perform calculation for every action α, completion stage c and time-step t. The algorithm can be easily adjusted to infer just some particular $X_{\alpha,c}$ and Y_t of interest, thus save computation cost.

A parameter to choose is the number of completion stages (which should not affect the inference complexity due to the reason explained above). It should be chosen big enough accordingly to the complexity of the primitive action. In the experiment of predicting human position using TUM dataset, we use 10 completion stages since the human movement is simple and smooth. In the experiment of predicting active hand using the toy assembly activity data, we have to use 100 completion stages since the movement of the hand during a primitive action can be fast with nonlinear trajectory.

D. Parsing and prediction in streaming mode

To do online inference, we apply a similar strategy as in previous sections: at each time-step, the primitive action detectors are run to update the factor tables, then the timings posterior are computed with the SIN framework. Then we run Kalman smoothing to compute the new value of $N(\hat{\mu_t}, \hat{\sigma_t})$ for every time-step t (while $\mu_{\alpha,c}, \sigma_{\alpha,c}$ are computed during training and stay the same during testing), finally the physical state posterior can be computed (figure 1). The posterior of Y_t for $t >$ current-time-step can be used for prediction. Note that the inference at each time-step is still independent of each other.

When running in streaming mode, available observations do not include the whole sequence. It is important to notice that $(\hat{\mu_t}, \hat{\sigma_t})$ are computed for every time-step t, using all available observation (and not necessary observation at time-step t). Hence we choose Kalman smoothing that can interpolate and extrapolate. Other dynamic systems such as [6] can be employed to better take advantage of environment visual feature if possible. This pre-processing step is practically estimation and prediction at low level.

On the other hand, parameters $\mu_{\alpha,c}, \sigma_{\alpha,c}$ of the factor F_{prior} encode high level information. Hence our method can be seen as the process of integrating these low level and high level information together.

V. Human position prediction in TUM Kitchen Dataset

The TUM Kitchen Dataset [25] contains 20 sequences of different subjects moving around the kitchen setting a table; different sensor data is provided: video, mocap, RFID tag and magnetic sensor reading. We experiment with 13 sequences where the activity is defined as "robotic", i.e. the sequence of actions is fixed and can be presented as a simple grammar. For this dataset, tracking the human is not difficult, so we will use mocap information as human position input and focus on the task of predicting the future position (predicting Y_t).

To apply the SIN framework, 2 components must be provided: the activity grammar and the primitive action detectors. We define the grammar with only 1 rule: the activity is an AND-composition of 14 primitive action. The human operator moves back and forth between the kitchen and the table to retrieve 7 objects. Figure 5 shows the Gaussian learnt for 10 completion stage of the primitive action [get-the-spoon]. For detecting primitive actions, we apply the learnt Gaussian distributions to the observed position of

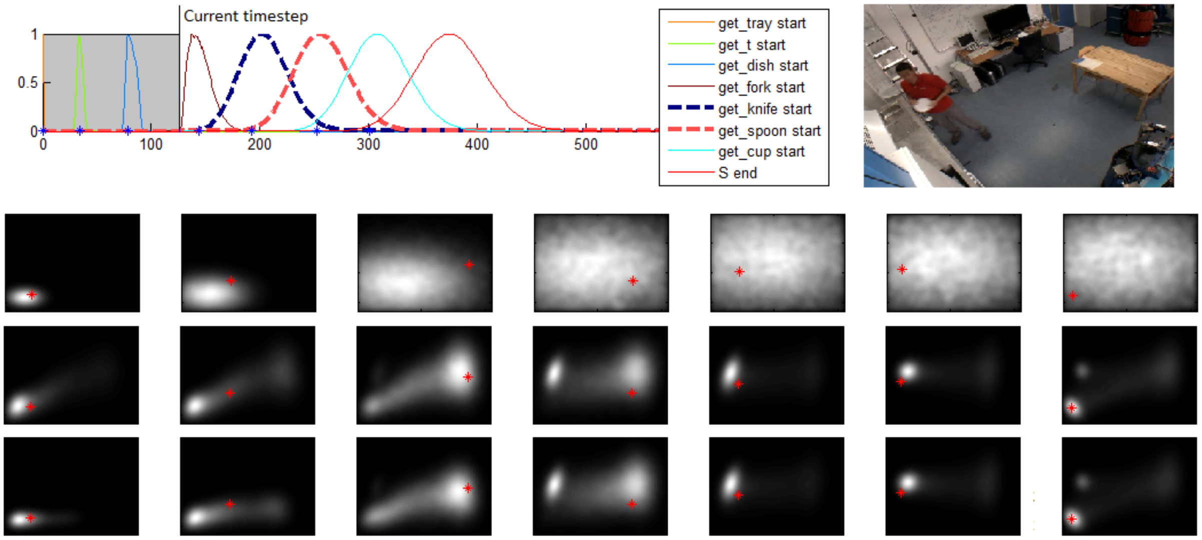


Fig. 7. (best view in color) Example of prediction in TUM dataset experiment running in streaming mode at time-step 135. The timing prediction result produced by SIN is on the top (not all is shown here), each curve represents a distribution and they were scaled to have the same height. The position prediction at 7 different points in the future (+0.5s, +1s, +2s, +4s, +8s, +16s, +32s) are at the bottom (the red marks are true position) for 3 methods: Without-F-prior (Kalman filter), Without-F-obv and our full model respectively. Observe that our method takes advantage of Kalman filter's dynamic model for short term prediction and activity temporal structure for long term prediction.

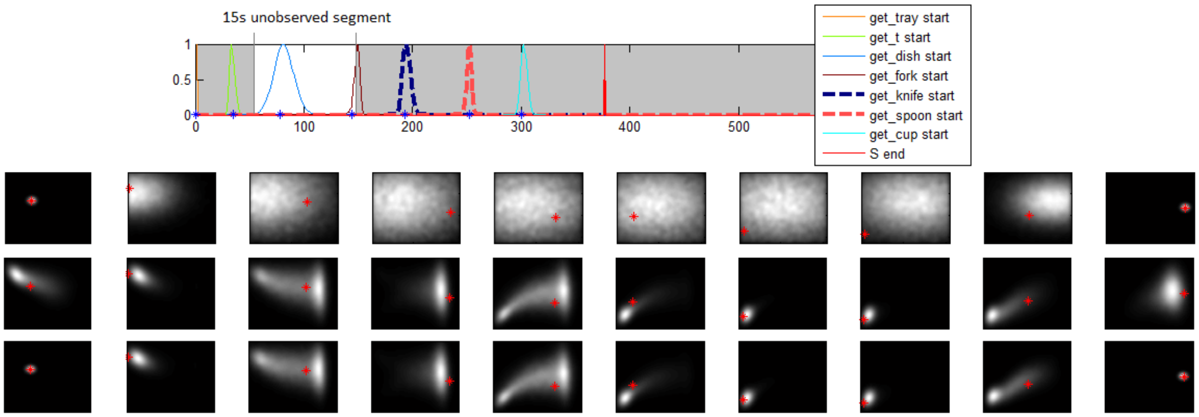


Fig. 8. Example of smoothing in TUM dataset experiment. The timing prediction result is on the top. The bottom show position estimation result of the 15s unobserved segment for 3 methods: Without-F-prior (Kalman filter), Without-F-obv and our full model respectively.

human during the activity and produce the likelihood score of that particular action (note that in general the detectors are allowed to incorporate other source of observation if it's available, not just restricting to the physical state only).

Our error measurement will be the average distance between the estimated position and the true position. When we run our method to parse the test sequence in streaming mode this estimation is always a future prediction. We choose 7 different points in time at which to predict the physical state: 0.5s, 1s, 2s, 4s, 8s, 16s, 32s in the future. At each time-step, inference is performed and the posterior distribution of the human position for those 7 different times in future is computed. For each distribution, we will use the mean as our estimated position and measure the error. We compute the average error over every time-step of every test sequence (in leave-one-out cross validation setting) and report them as prediction performance for each of those 7 different points in future.

Three baselines are used for comparison: (1) Prior position: the fixed position learnt from training data (which is the center of the kitchen), (2) Without F_{prior}: our method but the factor F_{prior} is disabled (i.e. set to uniform distribution), which is basically just constant velocity model Kalman filter prediction, and (3) Without F_{obv}: our method but the factor F_{obv} is disabled (i.e. set to uniform distribution).

The prediction result is shown in figure 6 (a). As expected, the further the future, the more ambiguous it is, hence bigger error. The error of the "prior" estimation (baseline 1) is high and mostly does not change. Kalman filter (baseline 2) is good for near future prediction, but it quickly gets worse. On the contrary, Baseline 3 (our method without F_{obv} from Kalman) accounts for the activity's temporal structure and the prior information of each action, hence it is better than Kalman filter in long term since past observation becomes less relevant to the far future. Our (fully) method can leverages both and perform well in both situations. A qualitative result of our prediction is shown in figure 7.

We also perform a smoothing task: a sequence is chosen for testing, the rest for training. The test sequence will be fully observed except for a 15s segment chosen at random. The task is infer the human position during the 15s unobserved segment. We use the same error measurement and baselines. Figure 8 shows an example result. Similar to the last experiment, our method outperforms the baselines. This trial is repeated 130 times and we report the average result in figure 6 (b).

VI. Hand position prediction in Toy assembly dataset

In this experiment, we will use the toy assembly activity data from our previous work [1]. It includes 29 sequences of a human operator assembling a toy model following a "recipe". This recipe is represented as a grammar, which has a total of 40 different primitive actions and 12 variations in the course of action. Each primitive action is to get a piece from one of the 5 bins in the workspace and assemble it. In [1], we show how to predict the timing (when each action is going to happen). We used a similar model in human-robot collaboration experiments [23], [24] where the robot assists human operator by predicting and preparing bins ahead of time. Here we will demonstrate how to predict the position of the active hand - the one reaching to the bins. We randomly select one sequence for testing and the rest for training.

The graphical model is constructed in the same manner as in [1] with the same grammar and primitive action detectors. We extend it with the set of variables X and Y to model the hand position. During training, the prior distribution of the hand at different completion stage for each primitive action are learnt. We use 100 completion stages for each primitive action. One example is shown in figure 9.

During test time, inference is performed to obtain the posterior of timings and hand position. In this experiment we did not measure the performance quantitatively since the hand is at the rest position most of the time. However qualitative result shows: (1) which actions might happen in the future and when; and (2) potential trajectories of the hand reaching for the bins corresponding to those actions. One example prediction result is shown in figure 10. The first one shows prediction result before the activity starts. The first 4 actions are reaching to bin 5, 5, 3 and 4 respectively ([Body] parts), hence we can see strong prediction of the hand reaching for bin 5 in the near future (in 5s) and for 3 bins 3, 4 ,5 in the far future (in 30s) (as it is not know when the activity is going to start). The other predictions are time-steps when [Nose_AB], [Tail_C] and [Wing_C] parts are assembled respectively. Observe that even the 5s in the future prediction can be ambiguous, this mostly due to the variation in the course of actions that can happen. For example the human operator can choose to assemble [Nose_AB] or [Nose_C] after assembling [Body] parts.

Next we show the prediction result at one particular time-step during an online run in figure 11. As more observations are available, the prediction result improves. Kalman smoothing helps prediction of the near future or smooth the hand trajectories after the fact since the detection might be noisy. The temporal structure help to make far future prediction and resolve overall ambiguity in the action sequence. Note that even when the reaching is detected, there's still ambiguity since many different actions are defined as reaching to the same bins. Only about 20s after the action started that is is identified.

VII. Conclusion

In this work, we considered the problem of making prediction of future state using knowledge about the temporal structure of the composite activity and the local dynamic of the state. We greatly extended our previous graphical model to incorporate these sources of information. As shown in the experiment, low level dynamic system such as Kalman filter performs well in very short term, while the learnt prior distribution of physical state in combination with timing reasoning is more useful, especially for far future prediction. Our model combines both in a unified framework. The output posterior density of the state reflects the system belief about the possible actions that will happen and the human movement during those actions.

References

[1] N. N. Vo, A. F. Bobick, From stochastic grammar to bayes network: Probabilistic parsing of complex activity, in: Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, IEEE, 2014, pp. 2641–2648.

[2] M. Pei, Y. Jia, S.-C. Zhu, Parsing video events with goal inference and intent prediction, in: Computer vision (iccv), 2011 ieee international conference on, IEEE, 2011, pp. 487–494.

[3] Z. Si, M. Pei, B. Yao, S.-C. Zhu, Unsupervised learning of event and-or grammar and semantics from video, in: ICCV, 2011.

[4] K. Li, J. Hu, Y. Fu, Modeling complex temporal composition of actionlets for activity prediction, in: ECCV, 2012.

[5] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, S. Srinivasa, Planning-based prediction for pedestrians, in: Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, IEEE, 2009, pp. 3931–3936.

[6] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, M. Hebert, Activity forecasting, in: Computer Vision–ECCV 2012, Springer, 2012, pp. 201–214.

[7] M. Huber, A. Knoll, When to assist?-Modelling human behaviour for hybrid assembly systems, in: Robotics (ISR), 2010.

[8] G. Hoffman, C. Breazeal, Cost-Based Anticipatory Action Selection for HumanRobot Fluency, IEEE Transactions on Robotics 23 (5) (2007) 952–961. doi:10.1109/TRO.2007.907483.

[9] G. Hoffman, C. Breazeal, Effects of anticipatory perceptual simulation on practiced human-robot tasks, Autonomous Robots 28 (4) (2010) 403–423.

[10] B. Ristic, S. Arulampalam, N. Gordon, Beyond the Kalman filter: Particle filters for tracking applications, Vol. 685, Artech house Boston, 2004.

[11] J. Yamato, J. Ohya, K. Ishii, Recognizing human action in time-sequential images using hidden markov model, in: Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on, IEEE, 1992, pp. 379–385.

[12] M. Brand, N. Oliver, A. Pentland, Coupled hidden markov models for complex action recognition, in: Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on, IEEE, 1997, pp. 994–999.

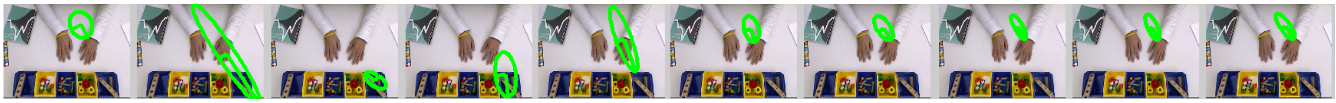


Fig. 9. Learnt distribution of hand position at the completion stages (1%, 10%, 20%, ... 100%) of 1 primitive action. This action involves the human operator taking a piece in bin 5 and assemble it.

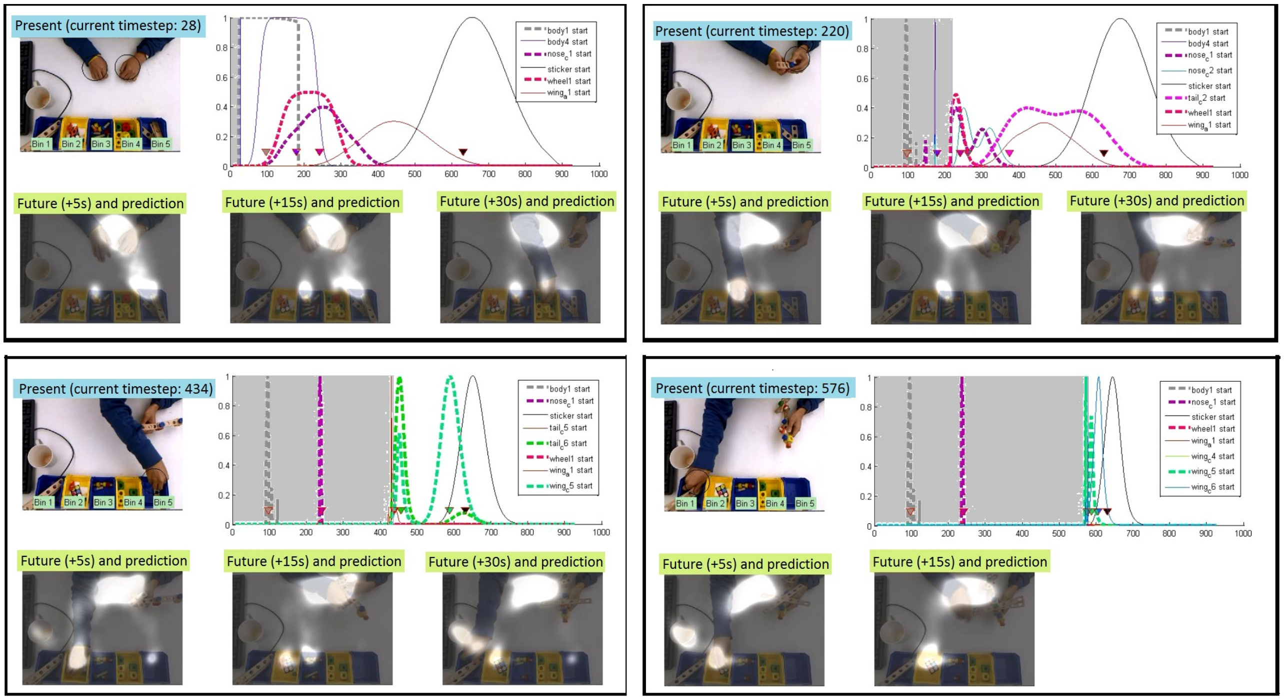


Fig. 10. Example of active hand prediction on 1 sequence at 4 different time-steps: 28, 220, 434, 576. At each time-step, we show the current frame image (top-left), the posterior of the timings (top right, not all distributions are shown, just the ones that are going to happen next) and future frame images (next 5, 10 and 30s) together with hand position prediction (bottom).

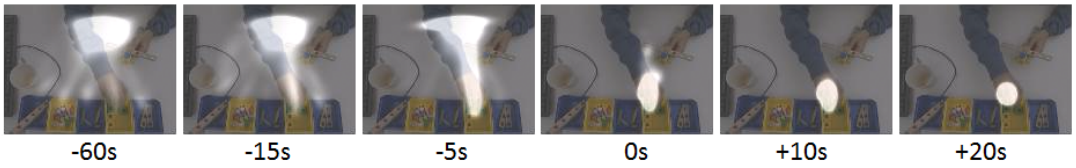


Fig. 11. Example of how the prediction improves in streaming mode. At this one particular time-step, the hand is reaching into bin number 4. From left to right are the prediction made 60s, 15s, 5s before the action starts, when the action starts, and 10s, 20s after.

[13] K. P. Murphy, Hidden semi-markov models (hsmms), unpublished notes.

[14] S.-Z. Yu, Hidden semi-markov models, Artificial Intelligence 174 (2) (2010) 215–243.

[15] K. Tang, L. Fei-Fei, D. Koller, Learning latent temporal structure for complex event detection, in: CVPR, 2012.

[16] V. Pavlovic, J. M. Rehg, J. MacCormick, Learning switching linear models of human motion, in: NIPS, Citeseer, 2000, pp. 981–987.

[17] S. M. Oh, J. M. Rehg, T. Balch, F. Dellaert, Learning and inferring motion patterns using parametric segmental switching linear dynamic systems, International Journal of Computer Vision 77 (1-3) (2008) 103–124.

[18] Y. Li, T. Wang, H.-Y. Shum, Motion texture: a two-level statistical model for character motion synthesis, in: ACM Transactions on Graphics (ToG), Vol. 21, ACM, 2002, pp. 465–472.

[19] Y. Shi, Y. Huang, D. Minnen, A. Bobick, I. Essa, Propagation networks for recognition of partially ordered sequential action, in: CVPR, 2004.

[20] B. Laxton, J. Lim, D. Kriegman, Leveraging temporal, contextual and ordering constraints for recognizing complex activities in video, in: CVPR, 2007.

[21] H. S. Koppula, A. Saxena, Learning spatio-temporal structure from rgb-d videos for human activity detection and anticipation, ICML, 2013.

[22] P. Wei, Y. Zhao, N. Zheng, S.-C. Zhu, Modeling 4d human-object interactions for event and object recognition, in: ICCV, 2013.

[23] K. P. Hawkins, N. Vo, S. Bansal, A. Bobick, Probabilistic human action prediction and wait-sensitive planning for responsive human-robot collaboration, in: Proceedings of the IEEE-RAS International Conference on Humanoid Robots, 2013.

[24] K. P. Hawkins, S. Bansal, N. N. Vo, A. F. Bobick, Anticipating human actions for collaboration in the presence of task and sensor uncertainty, in: Robotics and Automation (ICRA), 2014 IEEE International Conference on, 2014.

[25] M. Tenorth, J. Bandouch, M. Beetz, The TUM Kitchen Data Set of Everyday Manipulation Activities for Motion Tracking and Action Recognition, in: IEEE International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS), in conjunction with ICCV2009, 2009.