

# Context Identification for Efficient Multiple-Model State Estimation

Sarjoun Skaff                      Alfred A. Rizzi                      Howie Choset  
 sarjoun@ri.cmu.edu              arizzi@ri.cmu.edu              choset@ri.cmu.edu  
*Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA*

**Abstract**—This paper presents an approach to accurate and scalable multiple-model state estimation for hybrid systems with intermittent, multi-modal dynamics. The approach consists of using discrete-state estimation to identify a system’s behavioral context and determine which motion models appropriately represent current dynamics, and which multiple-model filters are appropriate for state estimation. This improves the accuracy and scalability of conventional multiple-model state estimation. This approach is validated experimentally on a mobile robot that exhibits multi-modal dynamics.

**Index Terms**—Hidden Markov Models, Timed Automata, Multiple-Model Filtering

## I. INTRODUCTION

Accurate and scalable state estimation for hybrid systems with intermittent dynamics is a key enabling technology for reactive control and system health monitoring. Examples of hybrid systems include legged mobile robots that exhibit different behaviors such as walking and jogging, and require robust state estimation for successful control. This paper presents a novel estimation approach that combines discrete and continuous state estimation techniques to accurately estimate the state of hybrid systems.

Conventional approaches consist of representing multi-modal dynamics with a collection of motion models, and performing state estimation with multiple-model (MM) filters [8], [9]. The MM approach associates multiple Kalman filters to each mode and runs all filters simultaneously. It averages the individual filters’ output weighted by their relative likelihood and generates a consolidated state estimate (Fig. 1). This approach requires the activation of the entire set of filters, which can become computationally intractable for robots that require a large number of models to accurately describe their dynamics.

Multiple-model filters can also produce inaccurate state estimates in the presence of dynamics that are not appropriately represented by available models. In realistic settings, unmodeled dynamics such as disturbances and transients can dominate locomotion dynamics and cause the failure of multiple-model estimation. The aim of this paper is not to model such disturbances, but rather to avoid using MM filters when unmodeled dynamics dominate.

To better understand these accuracy limitations, consider a generic hybrid system with the force profile of Fig. 2(a), and assume that available models appropriately represent the dynamic modes D1 and D2. The state can be estimated with multiple-model filters based on D1 and D2 models, as long as the system exhibits these dynamics. This is true only between the vertical dashed lines, where the system operates in steady state. Before and after that region, startup and stopping dynamics dominate and would cause the divergence

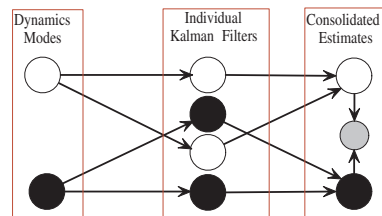


Fig. 1. Multiple-model systems deploy multiple Kalman filters for each mode of dynamics, weight the individual estimates by each filter’s relative likelihood, and consolidate the individual estimates into the system’s overall state estimate.

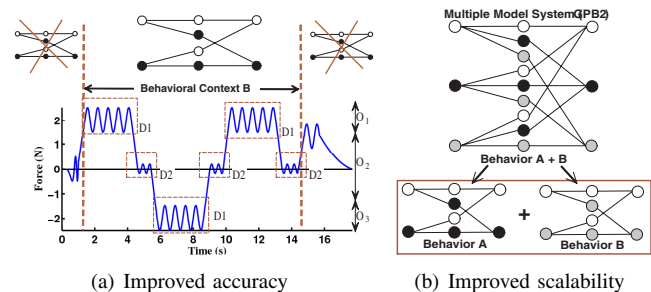


Fig. 2. The identification of a system’s behavioral context enables the deployment of multiple-model filters only when appropriate, which increases estimation accuracy (a). Behavioral context identification also increases estimation scalability by replacing large-scale MM systems with a collection of small-scale systems (b).

of the D1-D2 estimator. It is therefore necessary to verify that current dynamics are appropriately represented to generate accurate state estimates.

This paper presents an estimation framework that first identifies the dynamics and then determines which filters to use. The approach introduces the notion of *behavioral context* as representing the dynamics that are appropriately described by available models. This definition explicitly ties dynamics to the models that represent them, so identifying a robot’s behavioral context determines which models are appropriate for estimation. As a result, the framework prevents using inappropriate MM filters in the presence of unmodeled dynamics, as illustrated in Fig. 2(a), thus reducing the risk of estimation failure and improving accuracy.

With context identification, multiple-model estimators no longer need to activate all available filters, but only the filters that correspond to the current behavioral context. This reduces the computational overhead and increases the scalability of the estimation system. This concept is illustrated in Fig. 2(b), where a conventional large-scale multiple-model estimator designed for multiple behaviors is replaced with smaller-scale estimators specific to a single behavioral

context.

The context-based estimation framework uses discrete-state estimators to robustly identify behavioral contexts. The approach is based on the observation that hybrid locomotion dynamics induce spatial and temporal structure in the signal generated by onboard sensors. Recognizing these structures leads to the identification of the dynamics and their associated behavioral contexts. This is a pattern recognition problem solved in this paper by using hidden Markov models (HMMs) and timed automata to recognize spatial and temporal structures, thereby identifying the behavioral context.

The following sections provide a brief overview of related research in continuous- and discrete-state estimation, describe the context-based estimation framework, provide empirical examples of the limitations of conventional estimation, and demonstrate in experiment that context-based estimation improves the accuracy and scalability of multiple-model estimation systems.

## II. RELATED WORK

This section describes the discrete and continuous state estimation tools used by the context-based estimation framework, including multiple-model estimation, HMMs and timed automata.

### A. Multiple-Model Estimation

An accurate technique for estimating the state of hybrid systems is the generalized pseudo-Bayesian 2 (GPB2) algorithm [3], [7], a multiple-model estimator expressed in the Kalman filter framework. The computational complexity of the GPB2 grows quadratically with the number of models, as can be seen from a brief description of its estimation mechanism.

GPB2 estimation starts with evaluating hypotheses about the system being in one of its  $N$  dynamic modes at the previous sampling step, and transitioning into one of the same  $N$  modes at the current step. An individual Kalman filter based on the model of the current mode is associated to each hypothesis, so the total number of active filters (and of hypotheses) is  $N^2$ , a permutation among the  $N$  modes. The quadratic relationship compounded with the computational overhead of Kalman filtering limits the scalability of the GPB2. This makes the algorithm impractical for systems with a large number of models, and motivates the development of the context-based approach.

A three-model GPB2 cycle is shown schematically at the top of Fig. 2(b). The three initial hypotheses, represented with the first column of white, black and gray circles, spawn nine hypotheses and as many Kalman filters, represented by the middle column of nine circles. The last step is to prevent the exponential growth of the number of hypotheses by collapsing the nine hypotheses back to the original three (last column of three circles) [6]. At every iteration, a best state estimate can be extracted from the GPB2 by summing the output of all individual filters weighted by each filter's likelihood [3].

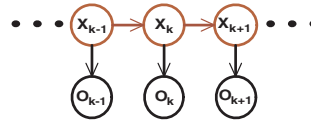


Fig. 3. Generic representation of hidden Markov models that transition among discrete states and generate discrete observations.

### B. Hidden Markov Models

An HMM is a probabilistic graphical model that undergoes transitions among its  $N$  states and generates discrete observations (Fig. 3). State estimation for an HMM is performed by computing a probability distribution  $\alpha_{k+1}(j)$  over its states  $i$  with the *forward* algorithm [10], which expresses the probability of the HMM process being in each of the states  $j$  (with  $\sum_{j=1}^N \alpha_{k+1}(j) = 1$ ) at step  $k+1$ :

$$\alpha_{k+1}(j) = \frac{\left[ \sum_{i=1}^N \alpha_k(i) \cdot a_{ij} \right] b_j(O_{k+1})}{\sum_{j=1}^N \left\{ \left[ \sum_{i=1}^N \alpha_k(i) \cdot a_{ij} \right] b_j(O_{k+1}) \right\}}, \quad (1)$$

where the transition probabilities  $a_{ij}$  express the probability of transitioning from state  $i$  at step  $k$  to state  $j$  at step  $k+1$ , and the observation probabilities  $b_j(O_{k+1})$  express the probability of generating the observation symbol  $O_{k+1}$  if the process is in state  $j$  at step  $k+1$ . Equation 1 is initialized with  $\alpha_1(i) = \pi_i b_i(O_1)$ , where  $\pi_i$  is the initial probability of state  $i$  and  $\sum_{i=1}^N \pi_i = 1$ . Transition and observation probabilities can be tuned manually or learned from labeled data using algorithms such as Baum-Welch or Expectation Modification [10].

The probabilistic expression of HMMs provides robustness to sensor noise, as current probability distributions mix observation probabilities  $b_j(O_{k+1})$  with model predictions  $\sum_{i=1}^N \alpha_{k+1}(i) \cdot a_{ij}$ . This property is used for robust recognition of a signal's spatial structure, as shown in the next sections, but not of its temporal structure, as HMMs do not explicitly model dwelling time in a state and cannot track a history of transitions among states. Capturing time could be performed by semi-Markov processes (SMPs), but SMPs have a high computational cost and are impractical for estimation [2], [14]. An alternative solution is to use timed automata.

### C. Finite State Automata

A finite state automaton (FSA) is a deterministic graphical model that undergoes input-triggered transitions among its states and generates discrete observations [4]. The context-based framework uses automata as a complement to HMMs, tracking transitions among HMM states and capturing in-state dwelling time. The automaton's estimation mechanism will be made clear in the example provided in the next section.

The main contribution of context-based estimation is the novel combination of discrete- and continuous-state estimation techniques for accurate and scalable estimation. Conventionally, classification and HMMs are used for speech recognition [5], visual object identification [12], action recognition

[13], and multiple-model estimation for aircraft fault detection and radar tracking [1], [9]. The context-based framework applies these techniques to robotic systems with intermittent dynamics.

### III. BEHAVIORAL CONTEXT IDENTIFICATION

Context-based state estimation applies principally to hybrid systems with cyclical dynamics, as the repetitive nature of the dynamics induce structure in sensor signal which is exploited to identify the behavioral context. The spatial and temporal components are defined as follows: *spatial* structure is the sequence of transition among salient points, or symbols, in a data stream; and *temporal* structure is the rate of transition among those symbols. The underlying assumption is that different dynamics induce distinct structures, so recognizing these structures uniquely identifies the dynamics and the behavioral context.

The description of the identification approach is carried step-by-step in the following subsections with the help of the generic hybrid system introduced earlier.

#### A. Hidden Markov Models

Consider again the force profile of Fig. 2(a), and recall that the behavioral context B corresponds to the steady-state region where D1-D2 filters are appropriate for estimation. The first step in identifying the context is to discretize the continuous force measurements into symbols that appear recurrently when the system is in context B. As shown in the figure, such sensor symbols can be O1, O2 and O3, roughly corresponding to the top, middle and bottom regions of the force profile, respectively.

The second step is to build a Markov chain model of a process that could generate these sensor symbols. A first model may contain the states H, M1, L and M2, generating the symbols O1, O2, O3 and O2, respectively (Fig. 4(a)). H and L correspond to high and low forces, M1 corresponds to medium forces resulting from transitioning from high to low forces, and M2 corresponds to low-to-high transitions. Obviously, the same symbol O2 is now generated by two different states M1 and M2, but the two states can be disambiguated by tracking the sequence of sensor symbols over time. If O2 appears after O1, then the system is in M1, and if it appears after O3, then it is in M2.

The third step is to specify HMM parameters as per Section II-B, with the transition and observation probabilities  $a_{ij}$  and  $b_j(O_{k+1})$  designed as to predict observation symbols in the expected sequence. For example,  $a_{M2|L} = 1$ ,  $a_{M2|j} = 0, \forall j \neq L$ ,  $b_{M1}(O2) = b_{M2}(O2) = 1/2$ , and  $b_H(O2) = b_L(O2) = 0$ . This enables the HMM to run the forward algorithm (Equation 1), process the symbols  $o$  extracted from data discretization and infer the probability distribution  $\alpha$  over the states.

When the system is operating in steady state, the observed symbol sequence is expected to match the sequence predicted by the model. Likewise, the sequence of HMM states<sup>1</sup> is expected to match the state transitions described by the

<sup>1</sup>In this paper, the sequence of HMM states is defined as the sequence of most likely states estimates by (1). At each step, the most likely state is  $\text{argmax}_{1 < j < N}(\alpha(j))$ .

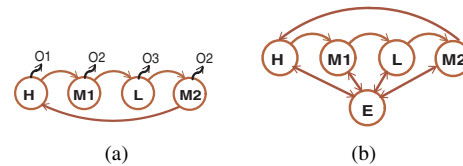


Fig. 4. HMM describing discrete process of a simple system.

model. When the dynamics vary from steady state, the sequence of symbols also varies from the predictions and leads to out-of-order state sequences. Therefore, verifying the order of state sequences helps recognize the behavioral context.

In order to enable the explicit detection of out-of-order transitions, the HMM model is augmented with an error state E, as in Fig. 4(b). The error state has low-probability, two-way transitions to all states and all observations have a uniform distribution over it; i.e., the error state is equally likely to generate all sensor symbols. The observation probabilities are designed such that the probability of observing a symbol conditioned on a state that should not generate it is lower than the symbol's probability conditioned on the error state. This means that the likelihood of generating a specific symbol by the error state is greater than the probability of generating that same symbol by a state that should not generate it. For example,  $b_E(O3) > b_H(O3)$ , and more generally,  $b_E(O_{k+1}) > b_j(O_{k+1})$  if  $O_{k+1}$  is different from the symbol generated by state  $j$ , where  $j$  is an index among the states.

This property ensures that if the wrong sequence of sensor symbols are observed, then the HMM would transition to the error state. For example, assume the model predicted that the system would transition to state  $j$ , but the observed symbol cannot be generated by  $j$ . This is an indication that the state sequence is out of order, so the observation probabilities ensure that  $\alpha(E) > \alpha(j)$ . In other words, the HMM assigns the highest likelihood to the error state E when the sequence of states is out of order. Thus, whenever the system is in E, the system is not expected to be in the behavioral context that corresponds to the HMM model (context B, in this case).

#### B. Finite State Automata

Detecting a single out-of-order transition is sufficient to recognize that the system is not operating in the expected behavioral context. However, contexts cannot be positively identified from observing a single in-order transition; a minimum number of transitions is necessary to avoid false positives.

Unfortunately, the HMM cannot track a sequence of transitions for longer than one time step, because of the Markov assumption. This motivates the use of finite state automata to perform bookkeeping. Here, the highest-likelihood HMM state is treated as inputs that trigger transitions among FSA states (Fig. 5). The FSA state structure is organized in  $p$  layers designed to recognize a correct sequence of inputs over  $p$  steps. Layers are defined as follows: the first layer contains the starting state; the final layer contains all the states reached after  $p$  numbers of correct transitions; and intermediate layers contain states reached after a number  $n$  of correct transitions, with  $n < p$ .

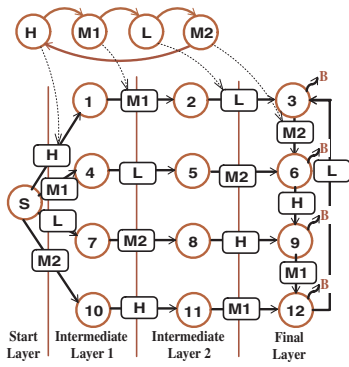


Fig. 5. HMM output such as H, M1, L, and M2 serve as input (rectangles) that trigger transitions between FSA states (circles). The structure of the FSA is organized in  $p$  layers in order track input sequences over  $p$  steps. The output flag ‘B’ indicates that context B is identified. Reset transitions back to S are omitted for clarity.

To see how this FSA can identify  $p$  successful transitions, assume that the initial input event transitions the automaton from the starting state  $S$  to a target state in the first intermediate layer. Consecutive occurrences of the same input cause self transitions, but new in-order inputs induce transitions to the next layer. After  $p$  correct transitions, the automaton reaches the final layer and outputs a success flag identifying the behavioral context (‘B’ in this case), and new in-order inputs maintain the automaton in the final layer. At any time, out-of-order inputs reset the automaton, and the context is no longer identified. In summary, success flags indicate that the signal’s spatial structure is recognized.

### C. Timed Automata

The FSA model designed to recognize the spatial structure of the signal can also be used to recognize the temporal structure. The temporal analysis mechanism turns the FSA into a timed automaton that measures delays separating consecutive inputs. Delays are measured by a clock reset each time the automaton enters a new state. Since HMM states correspond to automaton inputs, the automaton simultaneously measures its own and the HMM state duration.

Temporal information can be used as a timeout that triggers a transition out of a state if the duration exceeds a predefined bound. It also enables time-sensitive transitions, whereby the target state is selected as a function of both the input and the duration in the previous state.

## IV. EXPERIMENTAL RESULTS

Validation experiments are conducted on RHex [11], a six-legged dynamic robot able to walk and jog, among other behaviors. These experiments demonstrate that the HMM-timed automaton approach is successful at identifying behavioral contexts, and that contextual information improves the accuracy and scalability of MM estimators.

### A. Behavioral Context Identification

RHex alternates right and left tripods when it jogs, and its lateral acceleration while jogging is shown in Fig. 6. Available jogging motion models are only appropriate over

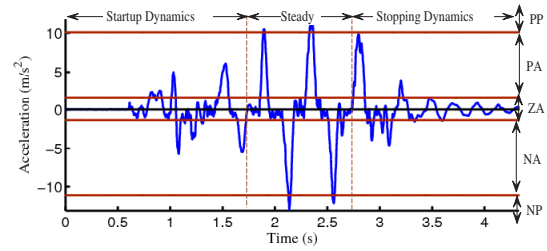


Fig. 6. Output of lateral accelerometer. The jogging gait induces positive and negative amplitudes during left and right tripod stances, and periods of near-zero accelerations during flight.

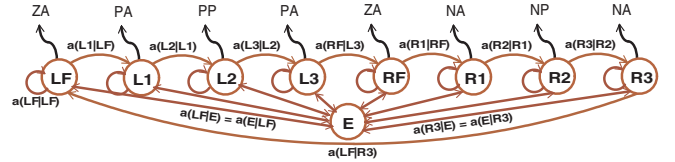


Fig. 7. Markov-chain model of the process generating jogging symbols. The error state is added to capture out-of-order state transitions.

the steady-state region indicated in the figure, so the task is to identify the robot’s jogging context in order to use jogging filters only when appropriate.

To this end, the acceleration data is first discretized as shown in the figure. The observation symbols are positive (PA), negative (NA) and zero (ZA) accelerations, and positive (PP) and negative (NP) peaks. PA corresponds to accelerations larger than  $1m/s^2$ , NA to accelerations smaller than  $-1m/s^2$ , and ZA spans the range in between.

A Markov chain model of the process generating these symbols is presented in Fig. 7. To accurately identify the jogging context, it is important to verify that jogging accelerations reach the expected amplitude, so the right and left tripods are represented by three states: the first state (R1 or L1) corresponds to the ascending acceleration, the second state (R2 or L2) corresponds to the peak acceleration, and the third state (R3 or L3) corresponds to descending acceleration. Thus, R1, R2 and R3 (or L1, L2 and L3) generate the symbols PA, PP and PA (or NA, NP and NA), respectively. LF corresponds to the flight phase during transitions from left to right tripods, and RF corresponds to flight during transitions from right to left, and E is the error state.

Transition probabilities are determined as follows. Self transitions  $a_{ii}$  to state  $i$  are computed by counting the number  $\bar{s}$  of symbols that occur when the system is in each state. The expected number of observations in a state, conditioned on starting in that state, is  $\bar{s} = \sum_{s=1}^{\infty} s \cdot p(s)$ , where  $p$  is the probability of undergoing  $s$  self-transitions to the same state. This equation can be re-written as  $\bar{s} = d(a_{ii})^{d-1}(1 - a_{ii}) = \frac{1}{1-a_{ii}}$  [10], so  $a_{ii} = 1 - 1/\bar{s}$ . For example, R2 generates on average 22 PP symbols, so  $a_{R2|R2} = 1 - 1/22 = 0.9432$ .

The transition probability from one state to the next is set equal to the transition probability from the same state to E, to ensure that the HMM detects out-of-order transitions. The probabilities for E are set as  $a_{E|E} = 0.999$ , and  $a_{E|i} = (1 - a_{E|E})/N$ , where  $N = 8$ , the number of states excluding E, and  $i \in N$ . The observations are set following the example of Section III-A and the probability distribution

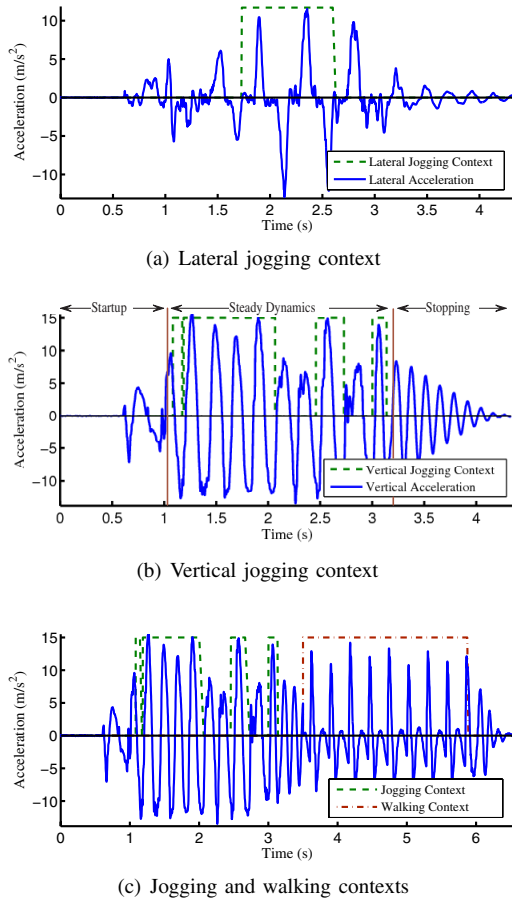


Fig. 8. Behavioral contexts identified along multiple dimensions and over multiple behaviors.

$\alpha$  is computed recursively with (1).

A timed automaton similar to Fig. 5 tracks the sequence of most likely HMM states over three layers, measures the in-state dwelling time, and outputs jogging flags when the jogging context is identified. The results of Fig. 8(a) show that the approach successfully recognizes the steady-state region over which jogging models can be used for estimation.

Identifying contexts along different dimensions is necessary when distinct motion models are used along each dimension. In this case, the same HMM-automaton technique successfully identifies behavioral contexts from acceleration measurements along RHex’s *vertical* axis, as shown in Fig. 8(b). The technique is shown in Fig. 8(c) to also enable accurate context identification for RHex’s walking behavior as well as for other behaviors that transition between walking and jogging. The variety of dynamical situations in which the technique is successfully applied provide empirical evidence of the approach’s robustness for context identification.

### B. Improving MM Scalability and Accuracy

Behavioral context identification is shown in experiment to improve the accuracy and scalability of multiple-model filters. Here, the task is to estimate RHex’s height from acceleration measurements generated while jogging, using GPB2 filters and jogging motion models.

The jogging behavior alternates flight and stance phases akin animal running, leading to oscillating accelerations shown in Fig. 8 (steady dynamics). The motion models used for the flight and stance phases are the ballistic projectile and the mass-spring system, respectively.

At the end of the experiment, the motors stop and the robot bounces on all six legs until it comes to rest. This induces damped oscillations of the stopping dynamics, referred to as the *stand* phase, which are represented with a mass-spring-damper model. In summary, the jogging behavior can be modeled with a collection of three models:

$$\ddot{z} = \begin{cases} -g, & \text{flight phase,} \\ -K_z(z - z_0)/M - g, & \text{stance phase,} \\ -2K_z(z - z_0)/M - g - (D/M)\dot{z}, & \text{stand phase,} \end{cases}$$

where  $z$  is the state representing the height of the robot;  $K_z$  the virtual spring constant of the mass-spring system;  $z_0$  the robot’s height at rest;  $M$  the robot mass;  $D$  a viscous damping parameter; and  $g$  gravity.

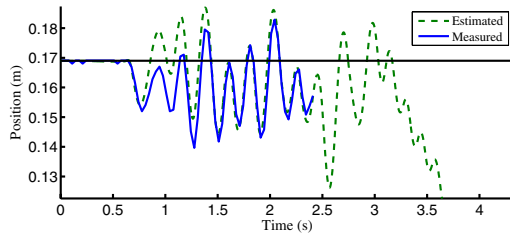
To estimate the height  $z$  of RHex, a conventional three-model, nine-filter GPB2 system is built, and its state estimates are plotted in Fig. 9(a). The plot shows that height estimates closely match ground truth measurements<sup>2</sup> over the steady-state region, but they diverge rapidly thereafter. Efforts to tune GPB2 and model parameters such as  $D$  could not improve the results reported. The cause of the divergence can be found in Fig. 9(b), which plots the acceleration before and after the motors stop at second 3. The plot reveals that the acceleration’s period of oscillation after stopping is shorter than the period before. The new periods prove too short for individual Kalman filters to converge, causing the filters to output incorrect likelihoods and the GPB2 to assign incorrect weights to individual estimates. Fig. 9(c) shows that after the motors stop, the GPB2 assigns the highest weight to the flight mode instead of the stand mode, which explains the downward slope of the height estimates.

This problem can be addressed by using the flight-stance models only over steady dynamics, and using the stance model when the robot is stopping. Information necessary to implement this strategy is provided by the contextual information of Fig. 8. This is done by assuming that the robot is either jogging or coming to rest and that motor state is unavailable, the strategy is to use a flight-stance GPB2 when the jogging context is identified, and a stance model when the robot is no longer jogging.

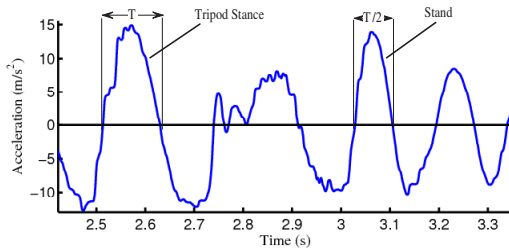
The results are reported in Fig. 10, where height estimates follow an oscillatory dissipative motion ending at the robot’s rest height of  $z_0$ . The ground truth measurement system’s limited workspace does not allow the quantitative validation of the estimate’s accuracy, but the fact that the estimates have the same frequency as the measured acceleration and that they converge to the rest height as expected is an indication of validity. Thus, this proof-of-concept example demonstrates that incorporating behavioral context information into MM filters improves accuracy by preventing divergence.

Incorporating context information also reduces computa-

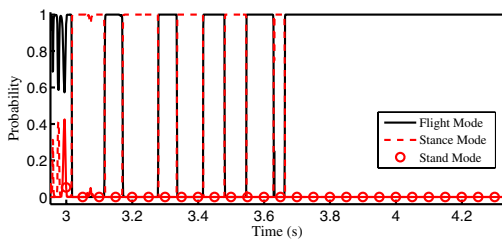
<sup>2</sup>The ground truth measurement system consists of high-speed cameras that register the position of LEDs placed on the robot body. The cameras cover a limited surface area, which explains the short experimental runs (about 4 seconds for jogging). Nevertheless, these results are useful because the available space is large enough to allow the robot to achieve steady-state motion before exiting the cameras’ field of view.



(a) Height estimates



(b) Oscillation periods. This plot zooms on the point of transition from steady-state to stopping dynamics.



(c) GPB2 mode probabilities. This plot zooms on the timeline for clarity.

Fig. 9. Conventional GPB2 estimation fails when individual filters do not have enough time to converge.

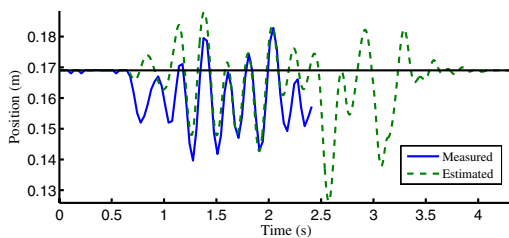


Fig. 10. Context information enables the selective activation of the jogging GPB2 and the stopping spring-damper model, and leads to accurate and scalable estimation.

tional requirements, as it enables the deployment of a two-model instead of the three-model GPB2, thus reducing the number of filters from nine to four and improving overall scalability.

## V. CONCLUSION

The context-based estimation framework enables accurate and scalable state estimation for systems with hybrid dynamics. This paper develops a novel approach of combining discrete and continuous estimation techniques to generate accurate estimates for systems with intermittent dynamics, in situations where conventional continuous-only approaches

fail. The paper also introduces the concept of explicitly tying motion models to the dynamics they represent, and demonstrates in experiment that this concept achieves significant accuracy and scalability gains for multiple-model filtering systems.

Behavioral contexts are an abstracted description of a robot's behavior, which provides high-level understanding of the robot's health (is the robot jogging as commanded?) and allows for closed-loop behavioral control.

Context-based estimation combines the conventionally separate fields of filter design and pattern recognition. Implementation examples provided in this paper show that simple approaches to pattern recognition (using HMMs) and to filtering (using KFs) lead to accurate estimates, which suggest that the designer does not need to acquire in-depth knowledge of each technique to generate satisfactory results.

Extensions will further improve the accuracy of discrete models by constructing HMMs that span multiple behaviors. In this configuration, the probability distribution  $\alpha$  over HMM states becomes a measure of confidence in each state, and by extension in the contexts themselves. As such,  $\alpha$  can be interpreted as a distance metric among behavioral contexts, measuring "how far" a robot's operation is from specific contexts. The distance metric can be used to close the loop on behavioral controllers, where a controller adjusts its parameters in ways that reduce distance to a target context, thereby improving control quality.

## REFERENCES

- [1] M. E. Campbell and S. Brunke. Nonlinear estimation of aircraft models for on-line control customization. In *IEEE Proceedings Aerospace Conference*, volume 2, pages 2/621 – 2/628, March 2001.
- [2] L. Campo, P. Mookerjee, and Y. Bar-Shalom. State estimation for systems with sojourn-time-dependent markov model switching. *IEEE Transactions on Automatic Control*, 36:238–243, February 1991.
- [3] C. B. Chang and M. Athans. State estimation for discrete systems with switching parameters. *IEEE Transactions on Aerospace and Electronic Systems*, 14(3):418–425, 1978.
- [4] Z. Kohavi. *Switching and Finite Automata Theory*, 2nd ed. McGraw-Hill, 1978.
- [5] Kai-Fu Lee. *Large-Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX System*. PhD thesis, Carnegie Mellon University, April 1988. Tech Report CMU-CS-88-148.
- [6] Uri Lerner. *Hybrid Bayesian Networks for Reasoning about Complex Systems*. PhD thesis, Stanford University, October 2002.
- [7] P. S. Maybeck. *Stochastic Models, Estimation and Control*, volume 1 of *Mathematics in science and engineering*. Academic Press, 1979.
- [8] P. S. Maybeck. *Stochastic Models, Estimation and Control*, volume 2 of *Mathematics in science and engineering*. Academic Press, 1982.
- [9] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan. Interacting multiple model methods in target tracking: a survey. *IEEE Transactions on Aerospace and Electronic Systems*, 34:103–123, January 1998.
- [10] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.
- [11] Uluç Saranlı. *Dynamic Locomotion with a Hexapod Robot*. PhD thesis, University of Michigan, September 2002.
- [12] A. Torralba, K. Murphy, W. Freeman, and M. Rubin. Context-based vision system for place and object recognition. In *The 9th International Conference on Computer Vision*, Nice, France, 2003.
- [13] J. Yamamoto, J. Ohya, and K. Ishii. Recognition human action in time-sequential images using hidden markov model. In *Proceedings of IEEE Conference Computer Vision and Pattern Recognition*, pages 379–385, June 1992.
- [14] Hakan Lorens Samir Younes. *Verification and Planning for Stochastic Processes with Asynchronous Events*. PhD thesis, Carnegie Mellon University, January 2005.