



Bayesian Hybrid Automata: A Formal Model of Justified Belief in Interacting Hybrid Systems Subject to Imprecise Observation

Paul Kröger ✉ 

Carl von Ossietzky Universität Oldenburg, 26111 Oldenburg, Germany

Martin Fränzle ✉ 

Carl von Ossietzky Universität Oldenburg, 26111 Oldenburg, Germany

Abstract

Hybrid discrete-continuous system dynamics arises when discrete actions, e.g. by a decision algorithm, meet continuous behaviour, e.g. due to physical processes and continuous control. A natural domain of such systems are emerging smart technologies which add elements of intelligence, cooperation, and adaptivity to physical entities, enabling them to interact with each other and with humans as systems of (human-)cyber-physical systems or (H)CPSes.

Various flavours of hybrid automata have been suggested as a means to formally analyse CPS dynamics. In a previous article, we demonstrated that all these variants of hybrid automata provide inaccurate, in the sense of either overly pessimistic or overly optimistic, verdicts for engineered systems operating under imprecise observation of

their environment due to, e.g., measurement error. We suggested a revised formal model, called Bayesian hybrid automata, that is able to represent state tracking and estimation in hybrid systems and thereby enhances precision of verdicts obtained from the model in comparison to traditional model variants.

In this article, we present an extended definition of Bayesian hybrid automata which incorporates a new class of guard and invariant functions that allow to evaluate traditional guards and invariants over probability distributions. The resulting framework allows to model observers with knowledge about the control strategy of an observed agent but with imprecise estimates of the data on which the control decisions are based.

2012 ACM Subject Classification Computer systems organization → Embedded and cyber-physical systems

Keywords and Phrases stochastic hybrid systems, Bayesian inference, formal models, cyber-physical systems

Digital Object Identifier 10.4230/LITES.8.2.5

Funding This research was supported by Deutsche Forschungsgemeinschaft under grant number DFG GRK 1765 covering the Research Training Group “SCARE: System Correctness under Adverse Conditions”.

Received 2020-10-01 **Accepted** 2021-11-16 **Published** 2022-12-07

Editor Alessandro Abate, Uli Fahrenberg, and Martin Fränzle

Special Issue Special Issue on Distributed Hybrid Systems

1 Introduction

Smart cities, automated transportation systems, smart health, and Industry 4.0 are examples of large-scale applications in which elements of intelligence, cooperation, and adaptivity are added to physical entities, enabling them to interact with each other and with humans as cyber-physical systems or, in the latter case, human-cyber-physical systems (CPSes or HCPSes). Due to the criticality of many of their application domains, such interacting cyber-physical systems call for rigorous analysis of their emergent dynamic behaviour w.r.t. a variety of design goals ranging from safety, stability, and liveness properties over performance measures to human-comprehensibility of their actions and absence of automation surprises. The model of hybrid (discrete-continuous)



© Paul Kröger and Martin Fränzle;

licensed under Creative Commons Attribution 4.0 International (CC BY 4.0)

Leibniz Transactions on Embedded Systems, Vol. 8, Issue 2, Article No. 5, pp. 05:1–05:27



Leibniz Transactions on Embedded Systems

LITES Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

automata [2, 28, 19], in its various flavours, has traditionally been suggested as a formal model accurately capturing CPS dynamics and thus facilitating such analysis with mathematical rigour whenever the pertinent requirements can also be formalised, which applies at least for the safety, stability, convergence, and liveness properties.

Hybrid automata (HA) provide a mathematical abstraction of the interaction between decision making, continuous control, and continuous environments. They couple a finite-state control skeleton with a continuous state-space spanned by real-valued variables. The continuous state has its dynamics governed by differential equations selected depending on the current control-skeleton state (often called a discrete mode or a control location), and vice versa state dynamics of the control skeleton is controlled by predicates on the continuous state. Various flavours of HA have been suggested as a means to formally analyse different aspects of hybrid-state dynamical systems, among them deterministic HA facilitating reasoning about their normative behaviour, non-deterministic HA [2, 28] under a demonic interpretation supporting worst-case analysis with respect to disturbances and measurement errors, and stochastic HA enabling quantitative verification [19, 31, 9, 15, 21, 14, 5]. Encoding the dynamics of an actual cyber-physical system into one of the aforementioned modelling frameworks is in general considered a tedious, yet mostly straightforward activity: it is assumed that these frameworks are rich enough to accommodate adequate models of standard components, like sensors measuring physical quantities and actuators modifying such quantities, as well as standard models of physical dynamics, continuous control, and mode-switching control.

In this article, which is an extended version of [17], we demonstrate that despite their embracing expressiveness and contrary to the intuition underlying the above modelling pragmatics, all flavors of hybrid automata fall short of being able to accurately capture the interaction dynamics of systems of well-engineered, rationally acting CPS designs operating under aleatory uncertainty. We show that the corresponding verification verdicts obtained on the best possible approximations of the actual CPS dynamics are across the range of hybrid automata models bound to be either overly optimistic or overly pessimistic, i.e., imprecise.

We identify inaptness to adequately cover rational decision making under uncertain information as the cause of this deficiency of the hybrid-automaton model. As such rational decision making requires manipulation of environmental state estimates to be embedded into the system state itself, necessitating manipulation of state distributions rather than “just” discrete plus real-vector valued state within the CPS and its corresponding formal model, we suggest an appropriate extension of hybrid automata featuring mixture-based probability distributions in some of its state variables. It adopts from metrology the concept of processing noisy measurements by means of filtering and representing the result as a distribution over possible ground truth [20, 29] and incorporates it into HA models. The resulting hybrid models can in general not be reduced to traditional HA featuring a finite-dimensional real-valued state vector, such that verification support remains an open issue that cannot be discharged by appropriate encoding into existing hybrid-automata verification approaches [13].

Organisation of the paper

In the subsequent section, we discuss related work in order to identify a current lack of models for hybrid dynamics being able to directly accommodate inference mechanisms about uncertain state observation. This would, however, not necessarily imply that current models are too weak for producing precise verdicts on system correctness, as an encoding of pertinent methods for fusing measurements could well be possible within existing models. In Sect. 3.2, we therefore demonstrate by means of a running example that traditional hybrid-system models are bound to fail in providing the expected verification verdicts. This in turn motivates us to introduce

filtering and state estimation into a revised model of hybrid automata. Section 3.3 demonstrates that this indeed leads to accurate verdicts adequately reflecting engineering practice, while Sect. 4 shows that an embedding of such environmental state estimation into traditional hybrid automata featuring real-vector state is in general impossible if the state estimation has to deal with states of other autonomous agents. Section 5 provides the formal definition of the suggested extension of hybrid automata before Sect. 6 puts forward ideas on automatic verification support for the resulting rich class of hybrid automata, and Sect. 7 concludes the paper by shedding light on related problems in the field of interacting intelligent systems.

2 Related work

An essential characteristic of cyber-physical systems is their hybrid discrete-continuous state-space, combining a continuous, real-vector state-space with a number of discrete modes determining the dynamics of the continuous evolution. Hybrid automata (HA) [2, 28] have been suggested as a formal model permitting the rigorous analysis of such systems. In their deterministic or demonically nondeterministic form, HA support qualitative reasoning in the sense of exhaustive verification or falsification, over the normative behaviour or the worst-case behaviour of the system. Probabilistic or stochastic extensions of HA, so-called stochastic hybrid automata [21], enable deriving quantitative figures about the satisfaction of a safety target by considering probability distributions over uncertain choices. Several variants of such a quantification have been studied, e.g., HA with discrete [31, 15] or continuous [14] distributions over discrete transitions as well as stochastic differential dynamics within a discrete mode [19].

HA models support the qualitative and quantitative analysis of systems subject to noise, yet lack pertinent means for expressing the effects of state estimation and filtering known to be central to rational strategies in games of incomplete information [25, Chapters 9-11] and thus in optimal control under uncertainty. Formal modelling of systems taking rational decisions based on best estimates of the uncertain and only partially observable state of other agents inherently requires to incorporate two levels of probabilism: first, in the model of system dynamics as probabilistic occurrences of sequences of observations; second, as distributions representing the best estimates the embedded controller can gain about the state of its environment based on these noisy observations. Formal modelling of rational decision making consequently requires the estimations to be explicitly available in the state space of the controller for evaluations of the underlying decisions (e.g., in the evaluation of a transition guard in supervisory control) and secondly correlated observations have to be fused to obtain best estimates, e.g. in form of Bayes filters [3, 22, 24]. Such probabilistic filters are widely used in robotics, e.g. for the estimation of occupancy grids [12, 7], in robust fault detection under noisy environment [6], or for estimating parameters of stochastic processes in biological tissues or molecular structures [30].

Aiming at approximating Maximum Likelihood Estimates for parameters of non-linear systems with non-Gaussian noise, Murphy [26] considers state estimation with switching Kálmán filters in presence of multiple linear dynamic models. In his setting, the time instances for switching to a certain linear dynamics are unknown up to a known stochastic distribution. In combination with stochastic state observations, this gives rise to state estimates in form of joint distributions, approximated by mixtures of Gaussian distributions. However, in addition to limited dynamics, switching between modes is based on Markovian dynamics, i.e., it is not possible to model switching based on probabilistic constraints on state estimates as necessary to model rational decisions about changing a mode as a response to observed states.

This lack of capabilities to model (rational) control decisions including discontinuous updates of the continuous state space is only partially resolved by the models underlying adaptive control theory, which is subject to comprehensive research [23, 18, 27]. In this context, the focus is on the identification of unknown (control) parameters of systems under imperfect observation. However,

these approaches are not sufficient to analyse the behaviour of interacting intelligent systems as they are restricted to identifying the correct choice between a set of (possibly time-variant) dynamical models for the controlled process.

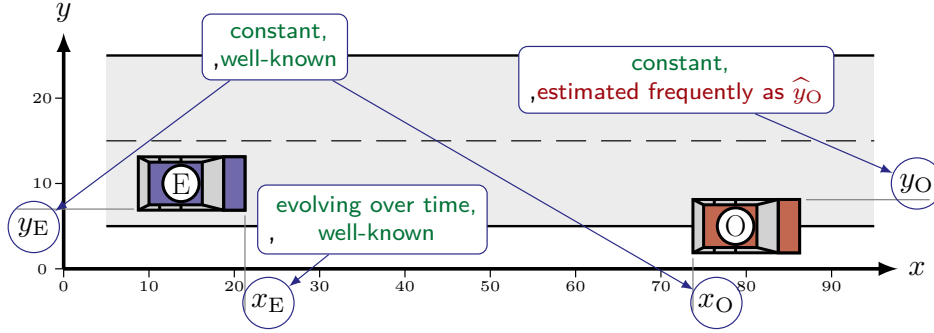
The consequential necessity of applying Bayesian filtering within hybrid systems implementing optimal control was already discovered by Ding et. al. [10]. They present an approach to derive optimal control policies for partially observable discrete time stochastic hybrid systems, where optimality is defined in terms of achieving the maximum probability that the system remains within a set of safe states. In order to be able to apply dynamic programming in search for an optimal solution, Ding et al. replace the partially observable system by an equivalent perfect information system via a sufficient statistics in form of a Bayes filter. This is very close to our approach in mindset, as a sufficient statistics about a Bayesian estimate of the imperfectly known actual system state is at the heart of rational decisions in control under uncertainty. The main difference is that we are trying to formulate a general model facilitating the behavioural analysis of such optimal hybrid control systems, while Ding et al. aim at the construction of such controllers w.r.t. a given safety goal. The latter facilitates a decomposition of the design problem into obtaining a Bayesian filtering process and developing a —then scalar-valued— control skeleton. This renders a direct integration, as pursued in this article, of state distributions and Bayesian inference mechanisms into the state space of an analytical model unnecessary.

In [16], we already suggested a revised formal model, called Bayesian hybrid automata, that is able to represent state tracking and estimation in hybrid systems and integrates probability density functions in its state space thereby enabling modelling of rational decision making under uncertain information. However, this model was not yet capable of covering hybrid-state dynamics of the observed agent when it comes to extrapolation of estimates over time between two measurement instances. As indicated in [17], this requires mixture distributions dealing with all possible decision alternatives (including the case that the decision is pending) in the state space of the model. In this article, we extend our previous work by a formal definition of an extension of Bayesian hybrid automata incorporating mixture distributions and a semantics covering hybrid-state dynamics for state-extrapolation.

3 Inadequacy of traditional hybrid-automata models

Hybrid automata have been conceived in [2, 28] as a formal model seamlessly integrating decision making with control, thus facilitating the modelling and analysis of the joint dynamics of these two layers pertinent to CPSes: discrete decisions, e.g. between the alternative manoeuvres of following a lead car or overtaking it in an autonomous car, do dynamically activate and deactivate continuous control skills, like an automatic distance control implementing the car-following manoeuvre. In HA, the former are described by a finite automaton featuring transitions guarded by (and possibly inducing side effects on) continuous state variables of the control path and the controller, while the latter are governed by differential equations attributed to (and thus changing in synchrony with) the automaton locations and ranging over the continuous state variables and thus describing the state dynamics of both the control path and the controller.

In reality, such CPSes have to operate and draw decisions under a variety of uncertainties stemming from their multi-component nature, as the latter requires mutual state observation between agents. Such sensing of non-local state inevitably induces uncertainties due to, a.o., the measurement inaccuracies inherent to sensor devices. In consequence, the decision making in real CPSes is bound to be rational decision-making under uncertainties. In this section, we demonstrate how existing hybrid-automata models fall short in taking account of such rational decision-making. To showcase the problem, we will in the following exploit a very simple example, mostly taken from [16], of a rational decision-making problem to be solved by a CPS.



■ **Figure 1** A common traffic situation (taken from [16]): ego vehicle E shall decide between passing the parked obstacle O or halting.

3.1 An example of a control decision problem

Our example deals with a common traffic situation depicted in Fig. 1. Our own autonomous car, called the ego vehicle and denoted by E in the sequel, is driving along a road which features another vehicle O parked further down the road. Despite being parked on the roadside, car O may extend into the lane used by E. E cannot perform a lateral evasive manoeuvre due to dense oncoming traffic. E therefore has to decide between passing the car while keeping its lane and an emergency stop avoiding a collision. It obviously ought to decide for a pass whenever that option is safe due to a small enough intrusion of O into the lane, and it should stop otherwise.

The geometric situation can be described by four real-valued variables: three rigid variables x_O , y_O , and y_E describing the static longitudinal position of O and the static lateral positions of both cars, as well as a flexible, continuously evolving variable x_E representing the momentary longitudinal position of the ego car. For simplicity, we assume that all values except the environmental variable y_O are exactly known to the ego car E. The value of y_O has to be determined by sensing the environment via a possibly inaccurate measurement yielding an estimate \hat{y}_O for y_O . For the sake of providing a concrete instance, we assume a normally distributed measurement error, i.e., $\hat{y}_O \sim \mathcal{N}(y_O, \sigma^2)$, though our findings do not hinge on that particular distribution. As a further simplification we assume that car E either drives with nominal speed ($\dot{x}_E = 1$) or is fully stopped ($\dot{x}_E = 0$) and that it switches between these two dynamics instantaneously.

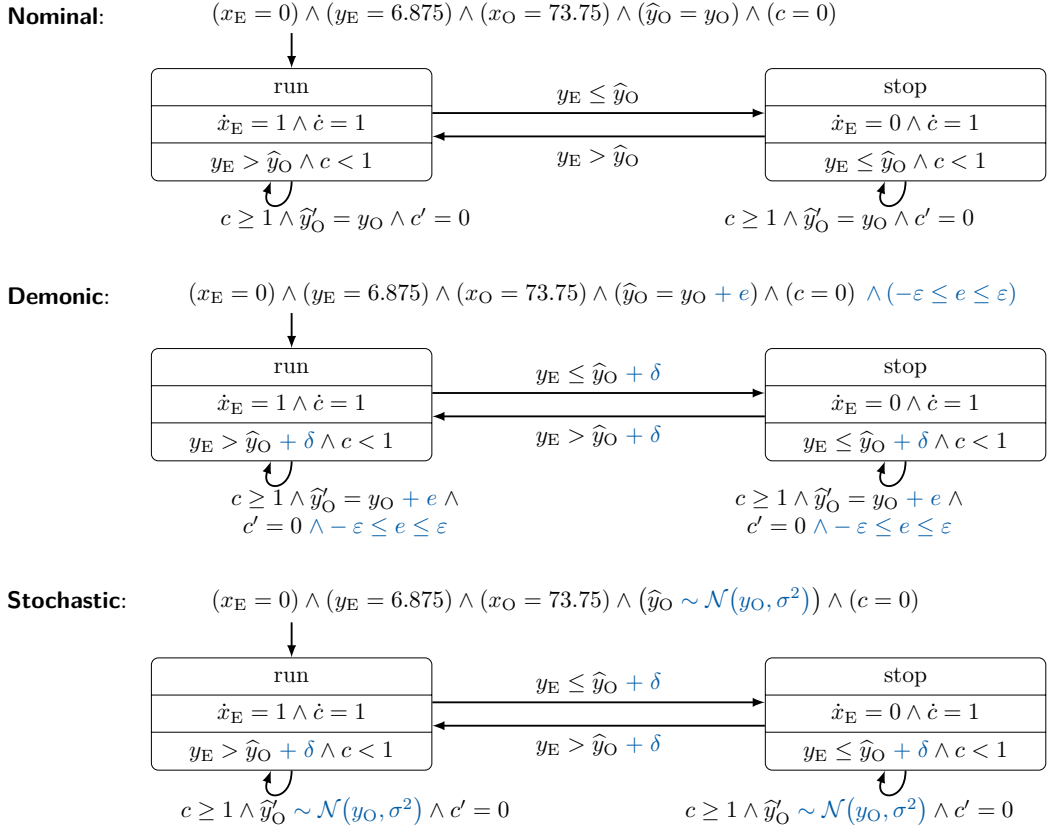
The design goal is to design an ego car that is both safe and live; the corresponding analysis goal consequently is to prove these two properties. Liveness in this context means that car E eventually passes car O whenever $y_E > y_O$. Safety is defined as the exclusion of the possibility of a collision, i.e., that $x_E < x_O$ stays invariant over time whenever $y_E \leq y_O$. These two properties can be formalised as follows using a straightforward extension of CTL featuring relational atoms over continuous signals akin to Signal Temporal Logic [11]:

$$\mathbf{safe} := (y_E \leq y_O) \implies \mathbf{AG}(x_E < x_O) \tag{1a}$$

$$\mathbf{live} := (y_E > y_O) \implies \mathbf{AF}(x_E \geq x_O) \tag{1b}$$

3.2 Hybrid automata models

Dealing with sensory observation of environmental variables and potentially reflecting the pertinent measurement inaccuracies within hybrid-automata models is a classical theme. Figure 2 represents the three standard means of dealing with sensory observation in HA models, exemplified on the example from the previous section: Automaton **Nominal** identifies environmental



■ **Figure 2** Hybrid automata models for the scenario of Fig. 1 (refinements of [16]). Variable c is a clock variable representing a timer that triggers a measurement every full time unit.

states with their measurements, thereby neglecting measurement error and claiming to draw control decision based on exact environmental entities. **Demonic** models measurement error as a bounded offset e between the actual value y_O and its measurement \hat{y}_O , with the offset e non-deterministically chosen afresh upon every take of a measurement. It also employs a safety margin δ within its decision making, passing only when the distance between y_E and \hat{y}_O is larger than the safety margin δ . **Stochastic**, finally, incorporates the faithful model of measurement noise by generating the measurement \hat{y}_O via a normal distribution $\mathcal{N}(y_O, \sigma^2)$ centred around y_O , where σ is the standard deviation of the measurement process.

Case analysis reveals that, depending on the relation between y_E and y_O and the safety margin δ , satisfaction of the two requirements formulae **safe** and **live** by the three models **Nominal**, **Demonic**, and **Stochastic** varies. Satisfaction applies as shown in Table 1.

None of these results seems particularly convincing. The nominal model, ignoring any measurement error in its analysis, optimistically claims its control to be both absolutely safe and live despite its decisions not even catering for adversarial measurement error impacting the non-robust guard $y_E > y_O$. The other two models pessimistically claim that it either is impossible to build any system satisfying any positive safety threshold ($P(\mathbf{safe}) \rightarrow 0.0$ in **Stochastic**) or to achieve any liveness (**Demonic** \neq **live**). Given that building such controllers and achieving very high quantitative degrees of, though not absolute, liveness and safety is standard engineering practice, all the above verdicts are disappointing and show inherent deficiencies in our conventional hybrid-state models.

■ **Table 1** Analysis results for the different models. $\rightarrow x$ denotes probabilities converging to x in the long-run limit.

(a) Analysis results for automaton **Nominal**.

	safe	live
$y_E > y_O$	trivial	sat
$y_E \leq y_O$	sat	trivial

Optimistic verdict, claiming perfect control possible despite the in reality inevitable uncertainty about environmental state.

(b) Automaton **Demonic** (case 3 arises only under insufficient safety margin $\delta < \varepsilon$ where $|e| \leq \varepsilon$).

	safe	live
$y_E > y_O + \delta + \max(e)$	trivial	sat
$y_O + \delta + \varepsilon \geq y_E > y_O$	trivial	unsat
$y_E \leq y_O < y_E - \delta + \varepsilon$	unsat	trivial
$y_E - \delta + \varepsilon \leq y_O$	sat	trivial

Pessimistic verdict, rightfully claiming safety at risk whenever an inappropriate safety margin is selected (case 3 in the table), but also claiming liveness perfectly impossible to achieve.

(c) Analysis results for automaton **Stochastic**.

	$P(\text{safe})$	$P(\text{live})$
$y_E > y_O$	1.0	$\rightarrow 1.0$
$y_E \leq y_O$	$\rightarrow 0.0$	1.0

Pessimistic verdict, claiming achievement of even marginal safety levels impossible over extended periods of time.

3.3 Adding Kálmán filtering

The obvious problem is that the above standard hybrid-automata models neglect the fact that repetition of noisy measurement processes accumulates increasingly better evidence about the true state of the observed entity, albeit always with a remaining uncertainty. While model **Nominal** ignores the impossibility of perfect knowledge, thus yielding inherently optimistic verdicts, models of the shapes **Stochastic** or **Demonic** do not correlate measurements across time series and thus fail to reflect the steady build-up of increasingly precise evidence about the true position y_O of the obstacle. Any form of truly rational decision-making would, however, take advantage of the latter fact; vice versa, any formal model neglecting it provides a coarse overapproximation of actual observational uncertainty resulting in correspondingly pessimistic verification verdicts relative to standard engineering practice employing filtering of measurements.

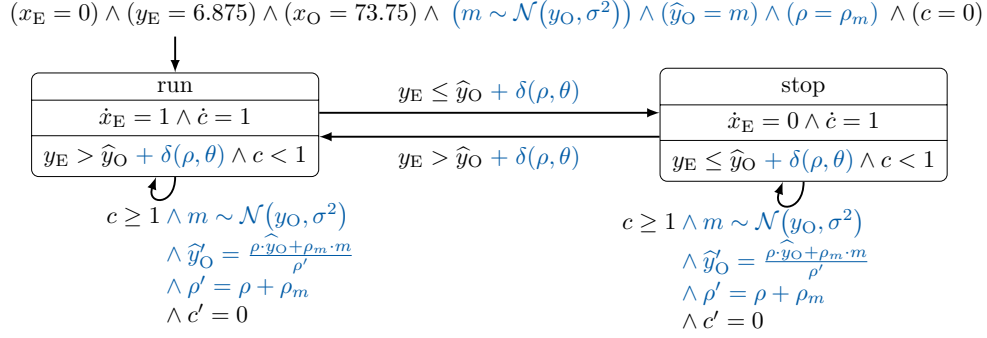
In the given case of a static obstacle O , as well as in the more general case of a physical process subject to purely linear differential dynamics, standard Kálmán filtering [20] manipulating normal distributions is the method of choice for obtaining best possible estimates of perceived state from independently normally distributed individual measurements. As normal distributions can be represented by a fixed number of parameters, namely their mean value and variance, these can still be incorporated into standard stochastic hybrid-automata models by means of extra variables: Retaining \hat{y}_O as the variable representing the current estimate of the lateral position of O in the scenario from Fig. 1, one has to add a second variable representing the accuracy of the current estimate. This could be the standard deviation or the variance of the estimation error; for simplicity of the update rules it is, however, customary to instead use the precision (i.e. the reciprocal of the variance). Adding a variable ρ representing the precision, the measurement transitions thus change according to the usual Kálmán-filter update rules

$$m \sim \mathcal{N}(y_O, 1/\rho_m) \tag{2a}$$

$$\hat{y}'_O = \frac{\rho \cdot \hat{y}_O + \rho_m \cdot m}{\rho'} \tag{2b}$$

$$\rho' = \rho + \rho_m \tag{2c}$$

where ρ_m is the precision of an individual measurement process and m the recent measurement.

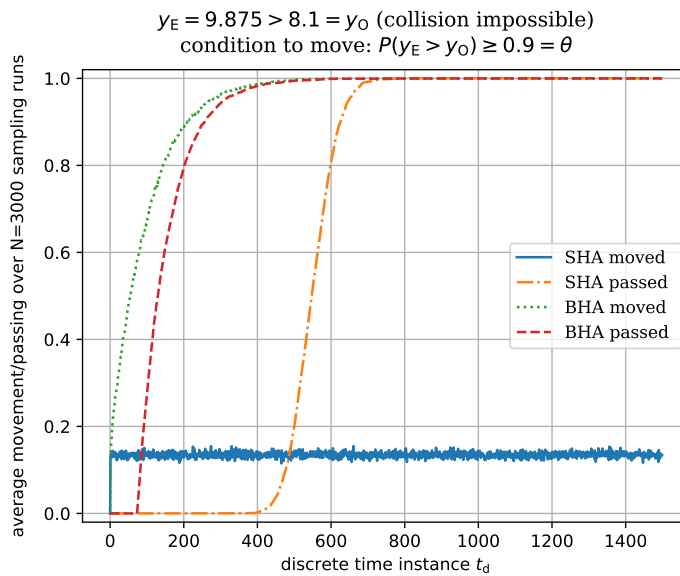


■ **Figure 3** A stochastic hybrid automaton incorporating Kálmán filtering for the measurements of O’s position. $\delta(\rho, \theta)$ computes a safety margin yielding confidence θ when the precision is ρ , i.e., is defined by $\int_{-\infty}^{\delta(\rho, \theta)} \mathcal{N}(0, 1/\rho) ds = \theta$.

The guards governing the decision to move to mode run (as well as the invariant of that mode) change to threshold conditions on the probability mass $P_{x \sim \mathcal{N}(\hat{y}_O, 1/\rho)}(y_E > x) \geq \theta$, checking for sufficient evidence that $y_E > y_O$ holds and thus confining the risk of erroneously moving the car forward when in fact $y_E \leq y_O$ applies to below $1.0 - \theta$. The resulting automaton is depicted in Fig. 3 and reflects the standard engineering practice of Kálmán filtering noisy measurements.

As can be seen from the experimental results reported in Figures 4 and 5, its control performance significantly exceeds all the verdicts for the standard models stated in Table 1. In these experiments, we implemented the automata **Stochastic** and its Kálmán-filtered variant (BHA) both in a safe situation (Figure 4) where car E is ought to pass car O, and in an unsafe situation (Figure 5) where car E should stop since O’s sphere overlaps with E’s lane. For both situations, the blue solid graph shows the average of switching to or remaining in mode run after a measurement (which is taken at every discrete time instance thereby assuming a step size of 1) for **Stochastic**. The average is constant for both situations. In contrast, the average of driving on converges to 1.0 rapidly for the safe situation while it converges fast to 0.0 for the unsafe situation (illustrated by the green dotted graph). This is where the Bayesian filter’s effect manifests itself: all decisions in **Stochastic** are based on the recent (single) measurement thus yielding a constant probability of making a “bad” decision as neither the distance $y_E - y_O$ nor the distribution of the measurement error changes over time. For the BHA, in turn, the integration of all measurement results leads to an estimate in form of a normal distribution of which the mean \hat{y}_O converges to y_O over time while the increasing precision allows for a less conservative safety margin.

The orange dashed-dotted line shows for each discrete time step the probability $P(x_E \geq x_O)$ in **Stochastic**, i.e. the probability that car E has already passed car O in the safe situation and that the cars have already collided in the unsafe situation. The red dashed line shows the same for BHA. As **Stochastic** moves with constant probability, the probability $P(x_E \geq x_O)$ of progressing beyond the other car’s position converges to 1.0 in **Stochastic** for both situations. This implies that car E almost surely eventually passes car O in the safe situation, but also almost surely eventually collides in the unsafe situation. These results were already predicted in Table 1. As a consequence of the effect of the filter, the graph for BHA shows for the safe situation that the probability that car E has passed car O increases significantly earlier than for **Stochastic**. Most probably, car E will pass car O quite smoothly after a short while in BHA, while it stutters past O in **Stochastic**. For the unsafe situation, in turn, the red dashed graph shows that the probability of a collision remains very small up to the time horizon.



■ **Figure 4** Simulation results for the traffic example (Fig. 1) comparing automaton **Stochastic** (labelled SHA) with its Kálmán-filtered variant (BHA) in a safe situation ($y_E > y_O$). BHA moves steadier (dotted green vs. solid blue line) and passes earlier (red vs. orange).

4 Interacting and cooperating cyber-physical systems

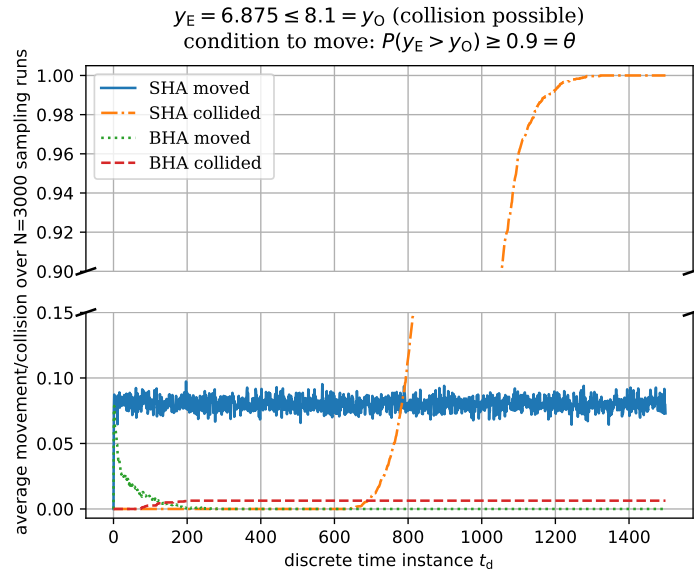
From the above, it might seem that encoding of standard engineering practice into stochastic hybrid automata well is feasible. Issues do, however, get more involved when the perceived objects are subject to more complex dynamics than linear differential equations s.t. normal distributions or other distributions representable by a finite vector of scalar parameters do no longer suffice for encoding optimal state estimates. This applies for example when the observed agent itself is a hybrid or cyber-physical system, as we will show in this next section. The above encoding into a stochastic hybrid automaton with finite-dimensional state becomes infeasible then, instead requiring to embed complex probability distributions directly into the automaton's state space.

To demonstrate this problem induced by the cooperation of smart entities, which hinges on the additional necessity to mutually detect and reason about control decisions of the mutually other agents based on uncertain behavioural observations, we now move on to a slightly more complex scenario involving interaction between cyber-physical systems.

4.1 An example of a cooperative control-decision problem

Imagine two ships approaching each other on a narrow channel permitting opposing traffic only within a designated passing place, as depicted in Fig. 6. The ship reaching the passing place first (ship O) is allowed¹ to draw a decision to which side it turns for mooring while the oncoming ship E enters the passing place. To complicate the issue, we forbid direct communication between the ships. In absence of means of negotiation, the ego ship E has to determine O's ensuing manoeuvre from observing the current lateral position of ship O and decides to move to a certain side as soon as its confidence that O will move to the opposite shore is above a specified threshold.

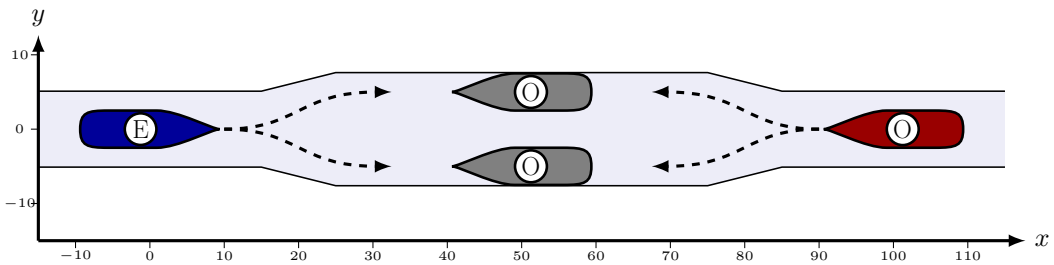
¹ Please note that this is a toy example ignoring all maritime rules such as COLREGs.



■ **Figure 5** Comparison in an unsafe situation ($y_E < y_O$) of the traffic example (Fig. 1). The Kálmán-filtered BHA enhances safety as it almost surely stops (dotted green line) and its collision probability saturates (dashed red), whereas the latter diverges for the SHA (dash-dotted orange) due to a constant rate of stuttering movement (solid blue).

We assume that ship O has perfect knowledge about its own longitudinal (x_O) and lateral (y_O) position. Ship E, in turn, has perfect knowledge about its own position (x_E and y_E) while it maintains estimates \hat{x}_O and \hat{y}_O of O's position. The problem for E is to determine, based on these estimates, to which side O will evade. Filtering w.r.t. a single known dynamics of O is no longer possible as dynamics depend on O's decision for which, in turn, E has only a probability distribution based on the estimate of O's position. Instead, mixture distributions dealing with all possible decision alternatives (including the case that the decision is pending) have to be dealt with. Each mixture component then covers the part of the state space of y_O that results in a certain decision and is subject to the corresponding dynamics within the filter process.

This obviously requires an extension of the stochastic hybrid automaton setting, as the estimates no longer constitute Gaussians due to the decision process itself, which is reflected by chopping the distributions at the thresholds of guards/invariants. That the underlying dynamics is non-linear only adds to the problem.



■ **Figure 6** Two ships approaching each other on a channel. The red ship (labelled O) decides to move to the right side of the passing place if its lateral position is larger than 0, and to the left otherwise. The ego ship (blue, labelled E) tries to determine O's manoeuvre and to move to the opposite side.

4.2 Formal modelling of the scenario

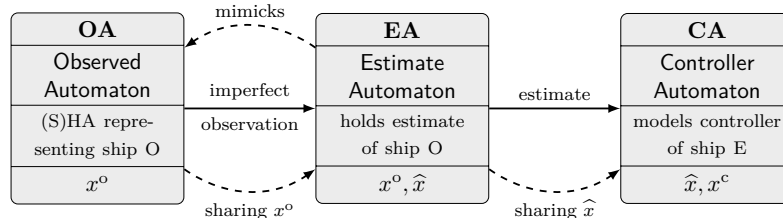
Given the complexity of the state estimation and rational decision processes sketched above, a decomposition of the overall problem into a set of interacting automata with dedicated functionalities seems appropriate. Figure 7 shows such a decomposition for the —still simple— case of unilateral observation, i.e., that ship O does not observe ship E and that their control behaviour consequently is not mutually recursive.

The roles of the various automata are as follows:

- *Observed automata (OA)* represent entities that are observed by the ego system. In the example, ship O is modelled by an observed automaton. As ship O has perfect knowledge about its own state (and as its behaviour is independent from E’s in the unilateral case), its automaton model is a traditional hybrid automaton of the same shape as **Nominal** in Fig. 2.
- *Estimate automata (EA)* provide estimates of O’s current state to E. They cover the perception process, i.e. they reflect the (possibly error-afflicted) environmental perception of the ego system and update quantitative estimates \hat{x} of the observed parameters x . In simple cases, they will regularly at sampling intervals take noisy copies $m \sim \mathcal{N}(x, 1/\rho_m)$ of the observed physical states and incorporate them into estimates \hat{x} for refine estimates \hat{x} . In addition, they extrapolate estimates over time between measurement intervals. The steps involved in creating and updating the estimates thus are manifold:
 1. Temporal extrapolation starts with splitting the current estimate, i.e., state distribution for the observed entity O according to O’s known mode selection dynamics. In the example, this would imply splitting the \hat{y}_O values of the part of the distribution that is associated to mode ‘run’ at 0 and associating its negative branch to mode ‘left’ and the non-negative to mode ‘right’.
 2. Reflecting possible sequences of instantaneous discrete jumps of O’s control automaton before time elapses, repeat step 1 until a fixed-point is reached which indicates that no further discrete jump is possible.
 3. For each mode, extrapolate these “fragments” of the distribution associated to the mode along the pertinent mode dynamics, which is followed for the duration of a single time step.²
 4. Take the resulting extrapolated distribution of O’s state, which now reflects the estimate of O’s state at the next measurement sampling time³, and pursue a Bayesian update of each of the individual fragments of the “dissected” distribution with a fresh measurement.

² For simplicity, we are assuming a discrete-time model here.

³ We assume that the size of the discrete time step equals the inter-sample time. Otherwise repeat from step 1 until the inter-sample duration is reached.



■ **Figure 7** Interplay between different automata modelling unilateral observation. Lowermost section lists types of variables accessed by the automata: x^o for system variables of the observed entity O, \hat{x} for state estimate variables associated to x^o within entity E, and x^c for system variables of the ego entity E.

05:12 Bayesian Hybrid Automata

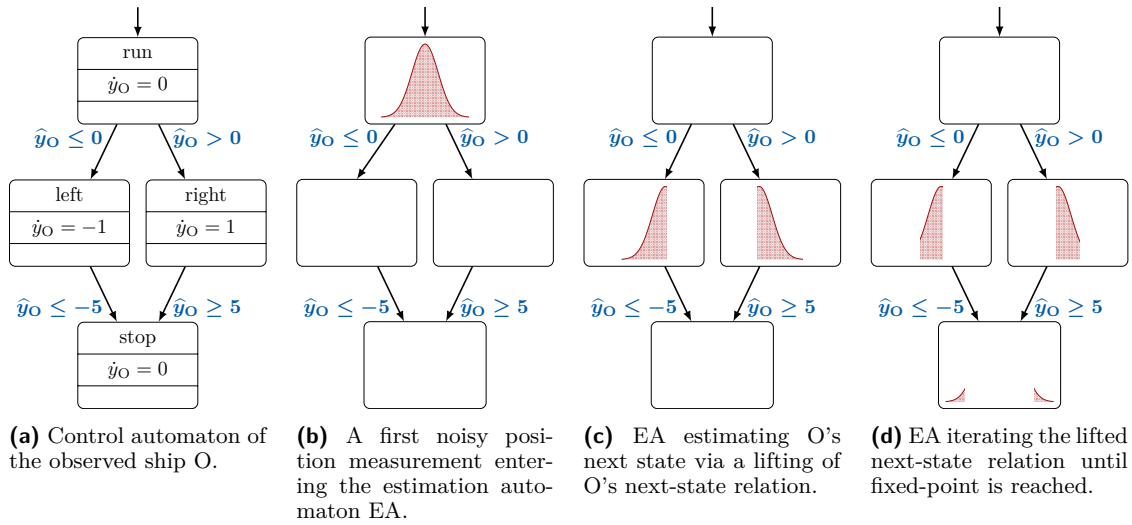


Figure 8 Estimating the observed ship's state within an estimation automaton: A noisy position measurement corresponds to a distribution of possible positions for O (b), each of which would drive O's control automaton (a) to a specific state. The EA reflects this by synchronously computing its estimate of O's hybrid state via a lifting of O's next-state relation to estimate distributions (c-d).

- Build the mixture of the resulting posterior "fragments" paired with their corresponding modes.

For a simple case, where the prior distribution merely stems from a single noisy measurement, steps 1 and 2 of the state extrapolation process are illustrated in Fig. 8b–8d.

- Controller automata (CA)** represent the controller of the ego system, i.e. of ship E in the example. Such a CA accesses estimate variables provided by EA if control decisions to be drawn involve estimated parameters. The corresponding decisions are "rational" in so far as safety-critical mode switches are based on sufficient confidence that the corresponding guard property is satisfied. Confidence here again relates to the probability that the guard condition gc holds true w.r.t. the estimated distributions: a critical transition is only taken if $P_{x \sim \mu}(gc) \geq \theta$, where θ denotes the required confidence and μ the mixture representing the current estimate of O's state. A safe alternative action (including a stay in the current mode iff its invariant bears sufficient evidence of being satisfied) has to be taken whenever no critical action can be justified with sufficient confidence.

For the sake of a concise presentation, we assume that there is a single measurement process or sensor for each observed parameter which is reflected within the estimate automaton. More differentiated observation processes are possible by, e.g., introducing another class of automata, possibly called *perception automata*, explicitly being responsible for measurements as suggested in [17]. Such an automaton could be a traditional stochastic hybrid automaton being located between OA and EA in Fig. 7 providing a noisy copy the observed physical state to the estimate automaton where the measurement process then depends on the perception automaton's internal state which, in turn, might change w.r.t., i.e., the internal state of the estimate automaton and the controller automaton.

Without digging into further detail of the above automata, it should be obvious that they go well beyond what can be encoded within hybrid automata models with their discrete plus finite-dimensional real-valued state-space:

1. As *controller automata* have to draw inferences about mixtures (representing state estimates) in order to evaluate their guards and invariants, such mixtures must be part of the state-space that controllers can observe.
2. As the state estimation by estimate automata involves active manipulation of such mixtures, these mixtures have to be part of their dynamic state.

State distributions therefore become first class members of the dynamic state themselves. As such state distributions can only rarely be encoded by finite-dimensional vectorial state (e.g., if they are bound to stay within a class of distributions featuring a description by a fixed set of parameters, like with normal distributions), this requires a completely fresh—and much more complex—set-up of the theory of hybrid automata extending beyond finite-dimensional vectorial state towards distributions as states. That this complication is necessary for obtaining accurate verdicts on control performance is witnessed by Figures 4 and 5.

5 Formal definition of the composite model

In the previous section, we presented a concept of formally modelling hybrid automata comprising Bayesian filter techniques for estimates of states of observed entities as well as mixture distributions representing those estimates. In this section, we introduce the formal models of observed automata, estimate automata, controller automata, and their combination into Bayesian hybrid automata. The resulting model is an extension of Bayesian hybrid automata suggested in [16] which were not yet capable of covering hybrid-state dynamics of observed entities within estimates.

5.1 Observed automaton

From an abstract perspective, an observed automaton might be an (almost) arbitrary flavour of traditional hybrid automaton. The assumption of perfect knowledge for observed automata as well as the assumption of unilateral observation allow to restrict the definition of observed automaton to deterministic variants. Aiming at a Gaussian character of estimates, we make two further assumptions which are a result of the fact that arbitrary continuous dynamics would lead to a “deformation” of probability density functions as well as arbitrary updates on discrete transitions would do. We hence assume that

1. the continuous dynamics of OA are constant for each mode, and
2. all updates of the continuous state of OA on a discrete transition is a shift by a constant.

In this article, we distinguish different types of Boolean predicates each of which represents a type of conditions on the continuous state space of a hybrid automaton enabling or disabling discrete control decisions, i.e. a type of transition guards and mode invariants. The first type is the traditional condition which essentially is equivalent to guard and invariant conditions of traditional hybrid automata.

► **Definition 1** (Traditional condition). Let \mathcal{X} be a set of n real-valued variables. A *traditional condition* is a predicate $c_t : \mathbb{R}^n \rightarrow \{\text{true}, \text{false}\}$. We denote the set of all traditional conditions by C_t .

We now define observed automata akin to the definition of Kowalewski et al. [21] with the restrictions mentioned above as follows:

► **Definition 2** (Syntax of observed automata). An observed automaton is a tuple $OA = (\mathcal{L}^o, \mathcal{X}^o, d^o, i^o, \Delta^o, g^o, u^o, \mathcal{I}^o)$ where

05:14 Bayesian Hybrid Automata

- $\mathcal{L}^\circ = \{\ell_1^\circ, \dots, \ell_p^\circ\}$ is a finite set of *discrete control modes* a.k.a. *control locations* of the automaton which is the discrete state space of OA ,
- $\mathcal{X}^\circ = (x_1^\circ, \dots, x_{n^\circ}^\circ)$ is an ordered finite set of *continuous system variables* conventionally represented as vector \underline{x}° and spanning the continuous state space of OA s.t. a pair $(\ell^\circ, \mathbf{x}^\circ) \in \mathcal{L}^\circ \times \mathbb{R}^{n^\circ}$ is a state of the automaton with $\mathbf{x}^\circ : \mathcal{X}^\circ \rightarrow \mathbb{R}$ being a variable valuation which is synonymously used for a concrete vector in \mathbb{R}^{n° ,
- $\mathbf{d}^\circ : \mathcal{L}^\circ \rightarrow \mathbb{R}^{n^\circ}$ is a *mode-dependent dynamics* defining the evolution of the continuous system variables \underline{x}° in relation to the control mode by specifying a differential equation $\dot{\underline{x}}^\circ = \mathbf{d}(\ell^\circ)$,
- $\mathbf{i}^\circ : \mathcal{L}^\circ \rightarrow C_t$ is a function describing the *invariants* per control mode, i.e. the part of the continuous state space for which OA may remain in the corresponding control mode,
- $\Delta^\circ \subseteq \mathcal{L}^\circ \times \mathcal{L}^\circ$ is a *discrete transition relation* between modes,
- $\mathbf{g}^\circ : \Delta^\circ \rightarrow C_t$ is a *guard function* decorating each discrete transition with a traditional condition defining the part of the continuous state space for which the corresponding transition is enabled s.t. Δ° is rendered deterministic,
- $\mathbf{u}^\circ : \Delta^\circ \rightarrow (\mathbb{R}^{n^\circ} \rightarrow \mathbb{R}^{n^\circ})$ is an *update function* decorating each discrete transition with a function $\mathbf{x} \mapsto \mathbf{x} + \underline{c}$ with $\underline{c} \in \mathbb{R}^{n^\circ}$ updating \underline{x}° when the transition is taken, and
- $\mathcal{I}^\circ \in \mathcal{L}^\circ \times \mathbb{R}^{n^\circ}$ is the initial state of OA .

We denote the set of all states of OA by Σ° .

► **Definition 3** (Semantics of observed automata). A run of an observed automaton is a sequence $\langle \sigma_0^\circ, \sigma_1^\circ, \dots \rangle$ of states $\sigma_i^\circ \in \Sigma^\circ$ with $\sigma_0^\circ = \mathcal{I}^\circ$ according to rule INITOA which is defined as

$$\frac{}{\sigma_0 = \mathcal{I}^\circ} \text{INITOA}$$

and for all $i \in \mathbb{N}^{>0}$ we have a transition $\sigma_{i-1}^\circ \xrightarrow{\text{STEP OA}} \sigma_i^\circ$ where the successor state is derived according to rule STEP OA which essentially is the concatenation of a discrete transition with a subsequent (discrete) time step. Rule STEP OA is defined as follows:

$$\frac{\sigma_{i-1} \quad \exists \sigma, \sigma_i \in \Sigma^\circ : \sigma_{i-1} \xrightarrow{\text{JUMPOA}} \sigma \xrightarrow{\text{TIME OA}} \sigma_i}{\sigma_i} \text{STEP OA}$$

Rule JUMPOA reflects a sequence of discrete jumps of OA . Since OA is deterministic and jumps are carried out instantaneously without consuming time, possible jumps enabled after a preceding jump have to be executed before a time step is possible. Hence, rule JUMPOA is basically the repeated application of taking a discrete transition, i.e. the repeated application of rule JUMPOA*, until a fixed-point is reached:

$$\frac{\begin{array}{c} \sigma \\ \exists \sigma^1, \dots, \sigma^k \in \Sigma^\circ : \sigma \xrightarrow{\text{JUMPOA}^*} \sigma^1 \xrightarrow{\text{JUMPOA}^*} \dots \xrightarrow{\text{JUMPOA}^*} \sigma^{k-1} \xrightarrow{\text{JUMPOA}^*} \sigma^k \\ \sigma^{k-1} = \sigma^k \end{array}}{\sigma^k} \text{JUMPOA}$$

Rule JUMPOA* reflects a single discrete transition of OA . If a discrete transition $(\ell^\circ, \ell^{\circ'})$ is enabled, the update of the continuous state maps \mathbf{x}° to $\mathbf{x}^{\circ'}$, and $\mathbf{x}^{\circ'}$ satisfies the invariant of $\ell^{\circ'}$, a jump from $(\ell^\circ, \mathbf{x}^\circ)$ to $(\ell^{\circ'}, \mathbf{x}^{\circ'})$ is possible:

$$\begin{array}{l}
(\ell^\circ, x^\circ) \\
(\ell^\circ, \ell^{\circ'}) \in \Delta^\circ \\
\mathbf{g}^\circ((\ell^\circ, \ell^{\circ'}))(x^\circ) \equiv \text{true} \\
\mathbf{u}^\circ((\ell^\circ, \ell^{\circ'}))(x^\circ) = x^{\circ'} \\
\mathbf{i}^\circ((\ell^{\circ'}))(x^{\circ'}) \equiv \text{true} \\
\hline
(\ell^{\circ'}, x^{\circ'}) \quad \text{JUMPOA}^*
\end{array}$$

Assume there is a solution $\underline{X} : [0, t] \rightarrow \mathbb{R}^{n^\circ}$ to the ordinary differential equation $d\underline{x}^\circ/dt = \mathbf{d}^\circ(\ell^\circ)$. If \underline{X} starts in x° and ends in $x^{\circ'}$ and all points in the image of \underline{X} satisfy the invariant of ℓ° , then a time step of length t from (ℓ°, x°) to $(\ell^{\circ'}, x^{\circ'})$ is possible:

$$\frac{(\ell^\circ, x^\circ) \quad \underline{X}(0) = x^\circ \quad \underline{X}(t) = x^{\circ'} \quad \forall t' \in [0, t] : \mathbf{i}^\circ(\ell^\circ)(\underline{X}(t')) \equiv \text{true}}{(\ell^{\circ'}, x^{\circ'})} \text{TIMEOA}$$

We assume $t \in \mathbb{R}^{>0}$ to be arbitrary but fixed.

The components of the observed automaton for our maritime example from Figure 6 could be defined as follows:

$$\mathcal{L}^\circ := \{\text{straight}^\circ, \text{left}^\circ, \text{right}^\circ, \text{stop}^\circ\} \quad (3a)$$

$$\mathcal{X}^\circ := (x_O, y_O) \quad (3b)$$

$$\mathbf{d}^\circ(\ell^\circ) := \begin{cases} [\dot{x}_O = -1, \dot{y}_O = 0]^T & \text{iff } \ell^\circ = \text{straight}^\circ \\ [\dot{x}_O = -1, \dot{y}_O = -1]^T & \text{iff } \ell^\circ = \text{left}^\circ \\ [\dot{x}_O = -1, \dot{y}_O = 1]^T & \text{iff } \ell^\circ = \text{right}^\circ \\ [\dot{x}_O = 0, \dot{y}_O = 0]^T & \text{iff } \ell^\circ = \text{stop}^\circ \end{cases} \quad (3c)$$

$$\mathbf{i}^\circ(\ell^\circ) := \begin{cases} x_O \geq 85 & \text{iff } \ell^\circ = \text{straight}^\circ \\ y_O \geq -5 & \text{iff } \ell^\circ = \text{left}^\circ \\ y_O \leq 5 & \text{iff } \ell^\circ = \text{right}^\circ \\ \text{true} & \text{iff } \ell^\circ = \text{stop}^\circ \end{cases} \quad (3d)$$

$$\Delta^\circ := \{(\text{straight}^\circ, \text{left}^\circ), (\text{straight}^\circ, \text{right}^\circ), (\text{left}^\circ, \text{stop}^\circ), (\text{right}^\circ, \text{stop}^\circ)\} \quad (3e)$$

$$\mathbf{g}^\circ(\delta) := \begin{cases} x_O \leq 85 \wedge y_O < 0 & \text{iff } \delta = (\text{straight}^\circ, \text{left}^\circ) \\ x_O \leq 85 \wedge y_O \geq 0 & \text{iff } \delta = (\text{straight}^\circ, \text{right}^\circ) \\ y_O \leq -5 & \text{iff } \delta = (\text{left}^\circ, \text{stop}^\circ) \\ y_O \geq 5 & \text{iff } \delta = (\text{right}^\circ, \text{stop}^\circ) \end{cases} \quad (3f)$$

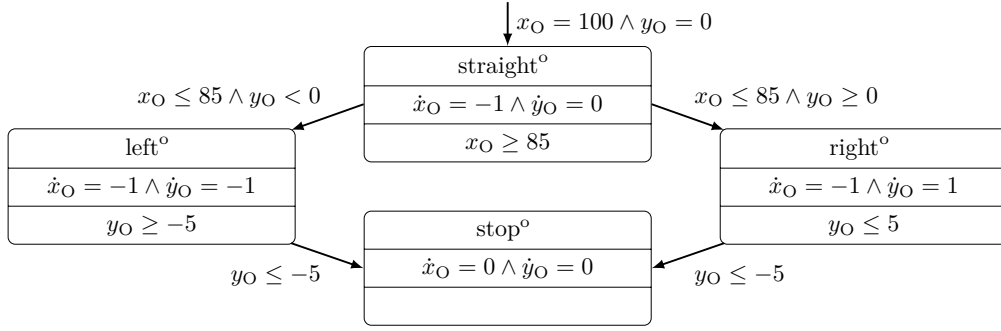
$$\mathbf{u}^\circ(x_O, y_O) := (x_O, y_O) \text{ for all } \delta \in \Delta^\circ \quad (3g)$$

$$\mathcal{I}^\circ := (\text{straight}^\circ, (x_O = 100, y_O = 0)) \quad (3h)$$

The automaton is illustrated in Figure 9.

5.2 Estimate automaton

Estimate automata govern the estimate of the observed system variables which are, essentially, a combination of information from a history of measurement results according to Bayes' theorem [29]. Estimate automata reflect the process of taking possibly error-afflicted measurements and applying the Bayes filter.



■ **Figure 9** Observed automaton for example from Fig. 6. Ship O switches to mode ‘left°’ if its lateral position is smaller than zero when entering the passing place, and to mode ‘right°’ otherwise. It stops when it reaches a shore.

An essential part of such a filter is to extrapolate the estimate along the continuous dynamics between two measurements. An estimate can be considered as a set of states each of which can be regarded to be the true state with some likelihood. Each of these states has to be evolved along the correct dynamics. Unfortunately, in the hybrid automata setting, this is not necessarily the same dynamics for all states as already indicated in Sect. 4.2: assume a set $A \subset \mathbb{R}$ enabling a discrete transition (ℓ, ℓ') while for $B = \mathbb{R} \setminus A$ the transition is not enabled. Then, the observed automaton may switch to ℓ' for all $x \in A$ while it has to remain in ℓ for all $x \in B$. Consequently, the dynamics of mode ℓ has to be applied to $x \in A$ as well as the differential equations of mode ℓ' govern extrapolations of trajectories starting in B .

This setting can be taken into account by interpreting estimates within estimate automata as a list of sets of continuous states annotated by their likelihood to be the true state in form of probability density functions as well as the control mode they are governed by. Such a list is then basically a mixture distribution where each mixture component is annotated by a control mode. In the example above, a discrete jump would lead to two mixture components \hat{x}_ℓ and $\hat{x}_{\ell'}$ for ℓ and ℓ' where the support of the (re-normalised) estimate \hat{x}_ℓ is restricted to A while the support of $\hat{x}_{\ell'}$ is restricted to B . We call such an extended mixture distribution a mixture estimate.

We now formally introduce components used by estimate automata including mixture estimates before providing the definition of the estimate automaton itself.

► **Definition 4** (Mixture estimate). A *mixture estimate* $\mu = (\mu_1, \dots, \mu_k)$ is a finite ordered set of mixture components where each component is an n -variate probability density function $\mu_i : \mathbb{R}^n \rightarrow \mathbb{R}$ and is labelled by an automata location and the weight of the component, i.e. we have labelling functions $\lambda_\ell : \mathcal{P}(\mathbb{R}^n) \rightarrow \mathcal{L}$ and $\lambda_P : \mathcal{P}(\mathbb{R}^n) \rightarrow (0, 1]$ s.t. $\sum_{i=0}^k \lambda_P(\hat{x}_i) = 1.0$ where $\mathcal{P}(\mathbb{R}^n)$ is the set of all probability density functions over \mathbb{R}^n .

A mixture estimate can be considered as a probability density function which is defined as the weighted sum of its components, i.e.

$$\mu(x) = \sum_{i=1}^k \mu_i(x) \cdot \lambda_P(\mu_i) \quad (4)$$

where $\sum_{i=1}^k \lambda_P(\mu_i) = 1.0$. We denote the set of all mixture estimates over \mathbb{R}^n by $\mathcal{M}(\mathbb{R}^n)$.

Uncertain conditions allow to model traditional conditions in the control strategy of the observed entity from the observer's perspective where a control decision is made with uncertainty since the continuous state satisfying or unsatisfying the corresponding condition is only estimated. However, they are rather a function than a predicate: for a given estimate $p \in \mathcal{P}(\mathbb{R}^n)$ and a traditional condition $c_t \in C_t$, an uncertain condition returns a normalised copy p' of p for which the support $\text{supp}(p')$ is restricted to those values satisfying c_t .

► **Definition 5** (Uncertain condition). An *uncertain condition* is a function $c_u : \mathcal{P}(\mathbb{R}^n) \times C_t \rightarrow \mathcal{P}(\mathbb{R}^n)$ with $c_u(p, c_t) \mapsto p'$ where the partial estimate over those values $x \in \mathbb{R}^n$ that satisfy c_t is defined as

$$p''(x) = \begin{cases} p(x) & \text{iff } c_t(x) \equiv \text{true} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

and lifted to a probability density function

$$p'(x) = p''(x) \cdot \frac{1}{\beta} \quad (6)$$

by re-normalisation based on the *branch probability*

$$\beta = \int p''(x) dx \quad (7)$$

which describes the probability that c_t is satisfied.

An uncertain jump then describes the effect of the discrete transition relation of the observed automaton to the observer's mixture estimate.

► **Definition 6** (Uncertain jump). Assume a hybrid automaton with a discrete transition relation Δ , a guard function g , and an update function u . An *uncertain jump* is a function $c_u^m : \mathcal{M}(\mathbb{R}^n) \rightarrow \mathcal{M}(\mathbb{R}^n)$ that takes a mixture estimate and applies the uncertain condition on all mixture components before the discrete jump of the continuous state space is applied to the resulting components, thereby generating a new mixture estimate:

$$\mu \mapsto \bigcup_{\mu_i \in \mu} \left(\bigcup_{\delta \in \Delta} \underbrace{u^o(\delta)(c_u(\mu_i, g^o(\delta)))}_{=\mu'_i} \right) \cup \underbrace{\left\{ c_u \left(\mu_i, \neg \bigvee_{\delta \in \Delta} g^o(\delta) \right) \right\}}_{=\mu''_i} \quad (8)$$

where μ'_i are those components obtained from following a discrete transition, μ''_i is the component obtained from that part of the continuous state space for which no transition is enabled, δ is an outbound transition of the mode annotated to μ_i (i.e. $\delta = (\ell, \ell') : \lambda_\ell(\mu_i) = \ell$), and $u^o(\delta)(\mu_j(x)) = \mu_j(2x - u^o(\delta)(x))$ is the lifting of the discontinuous update of the continuous state when a discrete transition is taken to probability density functions. The new mixture components are labelled with the target location of the corresponding transition, i.e. $\lambda_\ell(\mu'_i) = \ell'$ and $\lambda_\ell(\mu''_i) = \lambda_\ell(\mu_i)$ for the mixture component representing states remaining the in source location. Furthermore, each mixture component is labelled by a weight which is the probability that the new component is the true estimate, i.e. that the run of the observed automaton follows the sequence of discrete transitions from which the component is obtained. We consequently have $\lambda_P(\hat{x}') = \lambda_P(\mu_i) \cdot \beta$ (and $\lambda_P(\hat{x}'') = \lambda_P(\mu_i) \cdot \beta$, respectively) where β is calculated during evaluation of c_u according to Equation 7. In order to avoid a division by zero, components with $\beta = 0.0$ (hence components obtained from impossible discrete transitions) are simply dropped.

05:18 Bayesian Hybrid Automata

Perception is usually via sensors which, in general, is afflicted by errors. We model the process of drawing a measurement by a sensor function.

► **Definition 7 (Sensor).** Let $e \in \mathbb{R}^n$ be a random measurement error drawn according to an error distribution $\hat{e} \in \mathcal{P}(\mathbb{R}^n)$, denoted by $e \sim \hat{e}$. A *sensor* is a function $s : \mathbb{R}^n \rightarrow \mathbb{R}^n$ with $x \mapsto x + e$ and $e \sim \hat{e}$.

Bayes filters provide a recursive calculation of estimates combining “knowledge” from a sequence of measurements. In case of purely linear dynamics and a normally distributed measurement error, i.e. $e \sim \mathcal{N}(x, \sigma^2)$, a Kálmán-filter would be an instance of a Bayes filter yielding best estimates of the observed parameters. Such a dynamics and error model facilitates the implementation of concrete instances. However, as our findings do not hinge on that particular setup, we describe the application of such a filter in a very general manner by applying Bayes’ rule:

► **Definition 8 (Filter).** Let $r_k \in \mathbb{R}^n$ be the k -th measurement result obtained from a sensor s while $p_k(x) = p(x \mid r_k, \dots, r_1)$ is the estimate of $x \in \mathbb{R}^n$ after k measurements. Furthermore, \hat{r} is the conditional probability distribution $\hat{r}(r \mid x) = \hat{e}(r + x)$ of measurement results given x is the true parameter. By *filter* we denote a function $f : \mathcal{P}(\mathbb{R}^n) \times \mathbb{R}^n \rightarrow \mathcal{P}(\mathbb{R}^n)$ with

$$(p_{k-1}, r_k) \mapsto p_k(x \mid r_k, \dots, r_1) = \frac{\hat{r}(r_k \mid x) \cdot p_{k-1}(x \mid r_{k-1}, \dots, r_1)}{\hat{r}_{\mathbb{R}}(r_k)} \quad (9)$$

where $\hat{r}_{\mathbb{R}}(r_k) = \int \hat{r}(r_k \mid x^*) \cdot p_{k-1}(x^* \mid r_{k-1}, \dots, r_1) dx^*$.

Now that we have sensors and filters, we can define a measurement function as the combination of a sensor and a filter.

► **Definition 9 (Measurement action).** Let s be a sensor while f is a filter function. A *measurement action* is a function $m : \mathcal{M}(\mathbb{R}^n) \times \mathbb{R}^n \rightarrow \mathcal{M}(\mathbb{R}^n)$ with $(\mu, x) \mapsto \{f(\mu_i, r) \mid \mu_i \in \mu\}$ for a $r = s(x)$ fixed for all components μ_i of μ .

An estimate automaton is, essentially, a copy of the observed automaton augmented by measurement actions, estimate variables accommodating mixture estimates, and a filter function as well as a semantics in form of sequences of estimates.

► **Definition 10 (Syntax of estimate automata).** An *estimate automaton* is a tuple $EA = (\mathcal{L}^e, \mathcal{X}^o, \hat{\mathcal{X}}, m, d^e, i^e, \Delta^e, g^e, u^e, \mathcal{I}^e)$ where the elements annotated with e are copies of the corresponding elements of OA . The set \mathcal{X}^o of system variables is read-only shared with OA , i.e. a state change of \mathcal{X}^o in EA is directly passed through to \mathcal{X}^o in EA . Furthermore,

- $\hat{\mathcal{X}} = (\hat{x}_1, \dots, \hat{x}_{n^o})$ is an ordered finite set of *estimate variables* conventionally represented by a vector \hat{x} spanning the stochastic state space of EA s.t. \hat{x} is a state of EA where $\hat{x} : \hat{\mathcal{X}} \rightarrow \mathcal{M}(\mathbb{R})$ is the corresponding variable valuation specifying the marginal distributions of a mixture estimate in $\mathcal{M}(\mathbb{R}^{n^o})$ for which \hat{x} is synonymously used and \hat{x}_i is associated to x_i^o in the sense that \hat{x}_i accommodates an estimate of x_i^o in OA for all $i \in \{1, \dots, n^o\}$, and
- m is a *measurement action*.

We denote the set of all states of EA by Σ^e .

► **Definition 11 (Semantics of estimate automata).** A run of an estimate automaton is a sequence $\langle \hat{x}_0, \hat{x}_1, \dots \rangle$ of mixture estimates where \hat{x}_0 is deduced via rule INITEA which allows to derive the initial state of \hat{x} given the initial state $\sigma_0^o = \mathcal{I}^e$ of OA :

$$\frac{\sigma_0^o = (\ell_0^o, x_0^o) \quad r_0 = s(x_0^o) \quad \hat{x} = \mu(x \mid r_0) = \{\mu_1(x \mid r_0) = \hat{e}(r_0 - x_0^o)\}}{\hat{x}} \text{INITEA}$$

with $\lambda_\ell(\hat{x}) = \ell_0^o$ and $\lambda_P(\hat{x}) = 1.0$ where s is the sensor of m and \hat{e} is the error distribution of s .

For all $i \in \mathbb{N}^{>0}$ we then have a transition $\hat{x}_{i-1} \longrightarrow_{\text{STEP EA}} \hat{x}_i$ where the successor estimate is derived according to rule STEP EA which essentially is the concatenation of a discrete transition with a subsequent (discrete) time step:

$$\frac{\hat{x}_{i-1} \quad \exists \hat{x}^*, \hat{x}_i : \hat{x}_{i-1} \xrightarrow{\text{JUMPEA}} \hat{x}^* \xrightarrow{\text{TIMEEA}} \hat{x}_i}{\hat{x}_i} \text{STEP EA}$$

Rule JUMPEA is an abbreviation for two steps:

1. updating the mixture estimate via rule MEASURE and
2. applying the discrete dynamics via rule TRANSEA until a fixed-point is reached (akin to rule JUMPOA).

$$\frac{\begin{array}{c} \hat{x} \\ \exists \hat{x}^0 : \hat{x} \xrightarrow{\text{MEASURE}} \hat{x}^0 \\ \exists \hat{x}^1, \dots, \hat{x}^k : \hat{x}^0 \xrightarrow{\text{TRANSEA}} \hat{x}^1 \xrightarrow{\text{TRANSEA}} \dots \xrightarrow{\text{TRANSEA}} \hat{x}^{k-1} \xrightarrow{\text{TRANSEA}} \hat{x}^k \\ \hat{x}^{k-1} = \hat{x}^k \end{array}}{\hat{x}^k} \text{JUMPEA}$$

Rule MEASURE updates the estimate according by applying the measurement action m to each component of the mixture estimate:

$$\frac{x^o \quad \hat{x} \quad \hat{x}' = m(\hat{x}, x^o)}{\hat{x}'} \text{MEASURE}$$

Rule TRANSEA reflects the effect of OA 's control laws on the mixture estimate according to an uncertain jump.

$$\frac{\hat{x} \quad \hat{x}' = c_u^m(\hat{x})}{\hat{x}'} \text{TRANSEA}$$

The rule TIMEEA describes a discrete time step, i.e. the application of the continuous dynamics of OA (and EA , respectively) for t time units to the carrier of each mixture component:

$$\frac{\hat{x} \quad \hat{x}' = \{\mu_i(2\hat{x} - \hat{x}^*) \mid \mu_i \in \hat{x}\}}{\hat{x}'} \text{TIMEEA}$$

where for \hat{x}^* there is a solution $\underline{X} : [0, t] \rightarrow \mathbb{R}^{n^o}$ to the ordinary differential equation $d\underline{x}/dt = d^o(\lambda_\ell(\hat{x}))$ with $\underline{X}(0) = x$ and $\underline{X}(t) = x^*$ and $i^o(\lambda_\ell(x))(\underline{X}(t')) = \text{true}$ for all $t' \in [0, t]$. We assume $t \in \mathbb{R}^{>0}$ to be arbitrary but fixed.

For our maritime example from Figure 6, the estimate automaton would be a copy of the observed automaton defined in (3) extended by a set $\hat{\mathcal{X}}$ of estimate variables and measurement action m as follows:

$$\hat{\mathcal{X}} := (\hat{y}_O) \text{ with } \hat{y}_O = \{\mu_1, \dots, \mu_k\} \text{ as defined in Def. 4 and } \mu_i \text{ is a normal} \quad (10i)$$

distribution over \mathbb{R}

$$m(\hat{y}_O, r) := \{f(\mu_i, r) \mid \mu_i \in \hat{y}_O\} \text{ where } f \text{ is a Kálmán filter, and } r \text{ is a measurement} \quad (10j)$$

result obtained by a sensor s as defined in Def. 7 where the measurement error is normally distributed, i.e. $\hat{e} = \mathcal{N}(0, 1)$

Note that the main difference between observed automaton and estimate automaton is its interpretation, i.e. within the semantics (see Def. 11). Its graphical representation consequently is identical to Fig. 9.

5.3 Controller automaton

A controller automaton models the controller of the ego system which accesses the estimates of the estimate automaton in order to make rational decisions in the sense that mode switches based on estimated parameters are executed only in case of sufficient confidence that the corresponding guard property is satisfied. In this context, confidence is the probability that the guard property is satisfied w.r.t. the current mixture estimate, i.e. the probability distribution representing the belief about the state of the observed entity. We denote the class of predicates for such control decisions by “rational conditions”.

► **Definition 12** (Rational condition). Let \mathcal{X} be a set of n system variables while $\hat{\mathcal{X}}$ is the set of n estimate variables s.t. \hat{x}_i accommodates an estimate of the value of x_i . Furthermore, let $c_t \in C_t$ be a traditional condition. A *rational condition* is a predicate $c_r : \mathcal{M}(\mathbb{R}^n) \rightarrow \{\text{true}, \text{false}\}$ s.t.

$$c_r(\mu) \equiv \begin{cases} \text{true} & \text{iff } P_{\hat{\mathcal{X}}}(c_t \equiv \text{true}) \bowtie \varepsilon, \text{ and} \\ \text{false} & \text{otherwise} \end{cases} \quad (11)$$

with $\bowtie \in \{\geq, >\}$, $c_t \in C_t$, and $\varepsilon \in [0, 1]$ and

$$P_{\hat{\mathcal{X}}}(c_t \equiv \text{true}) = \int \mathbf{1}_{c_t}(\mathbf{x}) \cdot \mu(\mathbf{x}) \, d\mathbf{x} \quad (12)$$

where

$$\mathbf{1}_{c_t}(\mathbf{x}) = \begin{cases} 1 & \text{iff } c_t(\mathbf{x}) \equiv \text{true}, \text{ and} \\ 0 & \text{otherwise.} \end{cases} \quad (13)$$

Hence, a rational condition is satisfied iff the corresponding traditional condition c_t is satisfied with a probability larger than (or equal to) ε w.r.t. mixture estimate μ . We denote the set of all rational conditions by C_r .

As controller automata are not observed in the current setting, we can relax the restrictions we made for observed automata and allow non-determinism and arbitrary continuous dynamics and updates. However, for the sake of a more concise presentation, we perpetuate those restrictions.

► **Definition 13** (Syntax of controller automata). A *controller automaton* is a tuple $CA = (\mathcal{L}^c, \mathcal{X}^c, \mathbf{d}^c, \mathbf{i}^c, \Delta^c, \mathbf{g}^c, \mathbf{u}^c, \mathcal{I}^c)$ where

- $\mathcal{L}^c = \{\ell_1^c, \dots, \ell_{l^c}^c\}$ with $\mathcal{L}^c \cap \mathcal{L}^o = \emptyset$ is a finite set of *discrete control modes* a.k.a. control *locations* of the automaton which is the discrete state space of CA ,
- $\mathcal{X}^c = (x_1^c, \dots, x_{n^c}^c)$ with $\mathcal{X}^c \cap \mathcal{X}^o = \emptyset$ is an ordered finite set of continuous *system variables* conventionally represented as vector \underline{x}^c and spanning the continuous state space of CA s.t. a pair $(\ell^c, \mathbf{x}^c) \in \mathcal{L}^c \times \mathbb{R}^{n^c}$ is a state of the automaton with $\mathbf{x}^c : \mathcal{X}^c \rightarrow \mathbb{R}$ being a variable valuation which is synonymously used for a concrete vector in \mathbb{R}^{n^c} ,
- $\hat{\mathcal{X}} = (\hat{x}_1, \dots, \hat{x}_{n^o})$ is an ordered finite set of estimate variables as defined in Def. 10 and read-only shared with EA , i.e. a state change of $\hat{\mathcal{X}}$ in EA is directly passed through to $\hat{\mathcal{X}}$ in CA ,
- $\mathbf{d}^c : \mathcal{L}^c \rightarrow \mathbb{R}^{n^c}$ is a *mode-dependent dynamics* defining the evolution of the continuous system variables \underline{x}^c in relation to the control mode by specifying a differential equation $\dot{\underline{x}}^c = \mathbf{d}(\ell^c)$,
- $\mathbf{i}^c : \mathcal{L}^c \rightarrow C_t \times C_r$ is a function describing the *invariants* per control mode, i.e. the part of the continuous state space including the estimates for which CA may remain in the corresponding control mode,
- $\Delta \subseteq \mathcal{L}^c \times \mathcal{L}^c$ is a *discrete transition relation* between modes,

- $g^c : \Delta^c \rightarrow C_t \times C_r$ is a *guard function* decorating each discrete transition with a traditional condition or a rational condition defining the part of the stochastic state space of EA as well as the part of the continuous state space of CA for which the corresponding transition is enabled s.t. Δ^c is rendered deterministic,
- $u^c : \Delta^c \rightarrow (\mathbb{R}^{n^c} \rightarrow \mathbb{R}^{n^c})$ is an *update function* decorating each discrete transition with a function $x \mapsto x + \underline{c}$ with $\underline{c} \in \mathbb{R}^{n^c}$ updating \underline{x}^c when the transition is taken, and
- and $\mathcal{I} \in \mathcal{L}^c \times \mathbb{R}^{n^c}$ is the initial state of CA .

We denote the set of all states of CA by Σ^c .

► **Definition 14** (Semantics of controller automata). A run of an estimate automaton is a sequence $\langle \sigma_0^c, \sigma_1^c, \dots \rangle$ of states $\sigma_i^c \in \Sigma^c$ obeying deduction rules defined analogously to Def. 3 except for rules incorporating guards and invariants. We hence assume rules INITCA, STEPCA, and JUMPCA to be adopted straightforwardly from the semantics of the observed automaton where JUMPCA refers to JUMPCA* which respect both traditional and rational conditions for guards and invariants:

$$\frac{\begin{array}{l} (\ell^c, x^c) \\ \hat{x} \\ (\ell^c, \ell^{c'}) \in \Delta^c \\ \gamma((\ell^c, \ell^{c'}))(x^c, \hat{x}) \equiv \text{true} \\ u^c((\ell^c, \ell^{c'}))(x^c) = x^{c'} \\ \iota(\ell^{c'})(x^{c'}, \hat{x}) \equiv \text{true} \end{array}}{(\ell^{c'}, x^{c'})} \text{JUMPCA}^*$$

where

$$\gamma(\delta)(x^c, \hat{x}) \equiv c_t(x^c) \wedge c_r(\hat{x}) \quad \text{with } g^c(\delta) = (c_t, c_r) \quad (14)$$

and

$$\iota(\ell)(x^c, \hat{x}) \equiv c_t(x^c) \wedge c_r(\hat{x}) \quad \text{with } i^c(\delta) = (c_t, c_r) \quad (15)$$

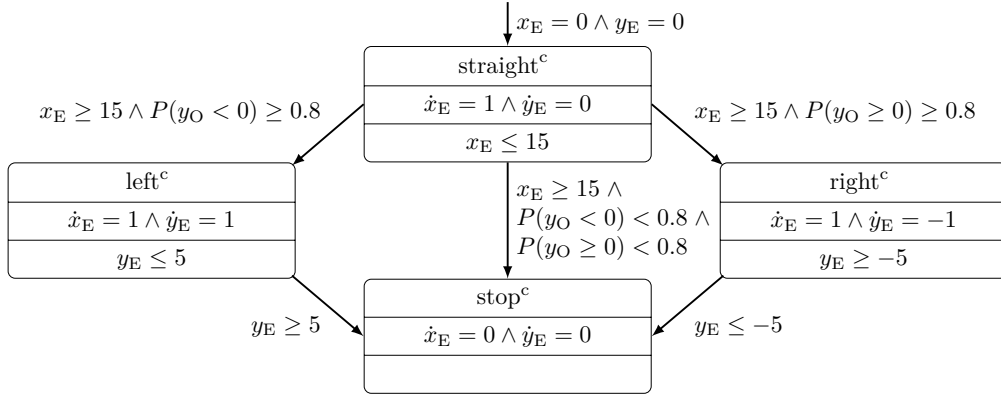
lift guard and invariant conditions to predicates $\mathbb{R}^{n^c} \times \mathcal{M}(\mathbb{R}^{n^o}) \rightarrow \{\text{true}, \text{false}\}$.

Analogously, rule TIMECA reflects a (discrete) time step using predicates $\iota(\ell)$:

$$\frac{(\ell^c, x^c) \quad \hat{x} \quad \underline{X}(0) = x^c \quad \underline{X}(t) = x^{c'} \quad \forall t' \in [0, t] : \iota(\ell^c)(\underline{X}(t'), \hat{x}) \equiv \text{true}}{(\ell^c, x^{c'})} \text{TIMECA}$$

for an arbitrary but fixed $t \in \mathbb{R}^{>0}$.

A controller automaton for ship E in our example from Fig. 6 is shown in Fig. 10. At position $x_E = 15$ it makes its decision to turn to a shore when it has sufficient confidence that for O the opposite shore is closer (and thus that O is turning to that shore). We assume that E performs an emergency stop if no sufficient confidence has been built up at that point. The corresponding automaton components are defined as follows:



■ **Figure 10** Controller automaton for the maritime example in Fig. 6. Ship E switches to mode ‘left^c’ when entering the passing place if it has sufficient confidence that ship O turns to the right shore (from E’s perspective), and to mode ‘right^c’ if it has sufficient confidence, that O turns to the left shore. E stops if none of the two options have sufficient confidence. It also stops when reaching a shore.

$$\mathcal{L}^c := \{\text{straight}^c, \text{left}^c, \text{right}^c, \text{stop}^c\} \quad (16a)$$

$$\mathcal{X}^c := (x_E, y_E) \quad (16b)$$

$$d^c(\ell^c) := \begin{cases} [\dot{x}_E = 1, \dot{y}_E = 0]^T & \text{iff } \ell^c = \text{straight}^c \\ [\dot{x}_E = 1, \dot{y}_E = 1]^T & \text{iff } \ell^c = \text{left}^c \\ [\dot{x}_E = 1, \dot{y}_E = -1]^T & \text{iff } \ell^c = \text{right}^c \\ [\dot{x}_E = 0, \dot{y}_E = 0]^T & \text{iff } \ell^c = \text{stop}^c \end{cases} \quad (16c)$$

$$i^c(\ell^c) := \begin{cases} (x_E \leq 15, \text{true}) & \text{iff } \ell^c = \text{straight}^c \\ (y_E \leq 5, \text{true}) & \text{iff } \ell^c = \text{left}^c \\ (y_E \geq -5, \text{true}) & \text{iff } \ell^c = \text{right}^c \\ (\text{true}, \text{true}) & \text{iff } \ell^c = \text{stop}^c \end{cases} \quad (16d)$$

$$\Delta^c := \{(\text{straight}^c, \text{left}^c), (\text{straight}^c, \text{right}^c), (\text{left}^c, \text{stop}^c), (\text{right}^c, \text{stop}^c), (\text{straight}^c, \text{stop}^c)\} \quad (16e)$$

$$g^c(\delta) := \begin{cases} (x_E \geq 15, P(y_O < 0) \geq 0.8) & \text{iff } \delta = (\text{straight}^c, \text{left}^c) \\ (x_E \geq 15, P(y_O \geq 0) \geq 0.8) & \text{iff } \delta = (\text{straight}^c, \text{right}^c) \\ (x_E \geq 15, P(y_O < 0) < 0.8 \wedge P(y_O \geq 0) < 0.8) & \text{iff } \delta = (\text{straight}^c, \text{stop}^c) \\ (y_E \geq 5, \text{true}) & \text{iff } \delta = (\text{left}^c, \text{stop}^c) \\ (y_E \leq -5, \text{true}) & \text{iff } \delta = (\text{right}^c, \text{stop}^c) \end{cases} \quad (16f)$$

$$u^c(x_E, y_E) := (x_E, y_E) \text{ for all } \delta \in \Delta^c \quad (16g)$$

$$\mathcal{I}^c := (\text{straight}^c, (x_E = 0, y_E = 0)) \quad (16h)$$

5.4 Bayesian hybrid automaton

► **Definition 15** (Bayesian hybrid automaton). A *Bayesian hybrid automaton* is a tuple $BHA = (OA, EA, CA)$ where

- OA is an observed automaton as defined in Def. 2,
- EA is an estimate automaton as defined in Def. 10, and
- CA is a controller automaton as defined in Def. 13.

A state of *BHA* is a tuple $\sigma^b = (\sigma^o, \sigma^e, \sigma^c)$ with $\sigma^o \in \Sigma^o$, $\sigma^e \in \Sigma^e$, and $\sigma^c \in \Sigma^c$. By Σ^b we denote the set of all states of *BHA*.

► **Definition 16** (Semantics of Bayesian hybrid automaton). A run of a Bayesian hybrid automaton is a sequence $\langle \sigma_0^b, \sigma_1^b, \dots \rangle$ of states which is obtained by a step of the observed automaton followed by a step of the estimate automaton and finally a step of the controller automaton. Hence, the initial state σ_0^b is derived by the following rule:

$$\frac{\frac{\vdash_{\text{INITOA}} \sigma^o \quad \{\sigma^o\} \vdash_{\text{INITEA}} \sigma^e \quad \vdash_{\text{INITCA}} \sigma^c}{(\sigma^o, \sigma^e, \sigma^c)}}{\text{INITBHA}}$$

For all $i \in \mathbb{N}^{>1}$ we have a step according to rule STEPBHA:

$$\frac{(\sigma_{i-1}^o, \sigma_{i-1}^e, \sigma_{i-1}^c) \quad \{\sigma_{i-1}^o\} \vdash_{\text{STEPOA}} \sigma_i^o \quad \{\sigma_{i-1}^e, \sigma_i^o\} \vdash_{\text{STEPEA}} \sigma_i^e \quad \{\sigma_{i-1}^c, \sigma_i^e\} \vdash_{\text{STEPCA}} \sigma_i^c}{(\sigma_i^o, \sigma_i^e, \sigma_i^c)} \text{STEPBHA}$$

5.5 Reduction of the mixture estimate size

The above semantics of estimate automata is based on, i.a., splitting probability density functions along discrete dynamics of the automaton. Thus the number of mixture components grows exponentially with the number of steps in the worst case. The reason for splitting and generating mixtures is to apply different continuous dynamics on specific parts of the continuous state space during extrapolation. We consequently can merge mixture components μ_i and μ_j by exploiting Eqn. (4) if they are labelled with the same discrete state, i.e., if $\lambda_\ell(\mu_i) = \lambda_\ell(\mu_j)$. In order to integrate such a merge of components, rule TRANSEA has to be modified s.t. it uses an *uncertain jump with merger* instead of the previously defined uncertain jump. The modified rule obviously requires that there exists a concise representation of the result which not explicitly itemises the individual summands. Such a representation is not inherently available.

► **Definition 17** (Uncertain jump with merger). Assume a hybrid automaton with a set of modes \mathcal{L} , a discrete transition relation Δ , a guard function \mathbf{g} , and an update function \mathbf{u} . An *uncertain jump with reduction* is a function $c_u^M : \mathcal{M}(\mathbb{R}^n) \rightarrow \mathcal{M}(\mathbb{R}^n)$ that applies an uncertain jump and reduces the number of mixture components according to Eqn. (4), i.e.

$$\mu \mapsto \bigcup_{\ell \in \mathcal{L}} \{\mu_\ell\} \tag{17}$$

where

$$\mu_\ell(x) = \left(\sum_{\mu_i \in M_\ell} \mu_i(x) \cdot \lambda_P(\mu_i) \right) \cdot \frac{1}{\lambda_P(\mu_\ell)} \tag{18}$$

with $\lambda_\ell(\mu_\ell) = \ell$ and the weight of the merged mixture component is

$$\lambda_P(\mu_\ell) = \sum_{\mu_i \in M_\ell} \lambda_P(\mu_i) \tag{19}$$

and the set of all mixture components in the result of an uncertain jump labelled by location ℓ is

$$M_\ell = \{\mu_i \mid \mu_i \in c_u^m(\mu) \text{ and } \lambda_\ell(\mu_i) = \ell\}. \tag{20}$$

6 Automated verification

While there is a strong body of research concerning state-exploratory methods for hybrid-system verification or falsification (for an overview cf. [13]), all of these methods are currently confined to finite-dimensional, hybrid discrete-continuous state. With complex, mixture-type distributions becoming part of the state-space of the system itself, these methods are no longer applicable: State-exploratory methods for stochastic hybrid automata [31, 15, 1, 14] do, of course, manipulate complex mixtures over hybrid state, but as these stem from the stochastic transition dynamics rather than from the state-space itself, state-exploratory methods covering estimate automata inherently have to add another layer of mixtures (due to the iterated transition dynamics of estimate automata) ranging over state mixtures (themselves being part of the state of estimate automata). To the best of our knowledge, neither a comprehensive tool nor data structures facilitating such an analysis do currently exist.

Simulation faithfully reflecting the arising distributions in a frequentistic sense, however, seems feasible. This would facilitate rigorous statistical model-checking [32]. As mixtures of hybrid states are part of the state-space itself due to the hybrid-state estimation process (see Fig. 8), the underlying simulator, however, has to manipulate the corresponding mixtures as part of its state-space.

If all mixtures arising are linear combinations of interval-restrictions of normal distributions, as in Fig. 8, a length-unbounded list of such interval-restrictions suffices for representing the mixture part of the state space, which then has to be combined with the usual discrete and real-vector-valued states of hybrid automata. Each individual interval restriction of a (scaled) normal distribution can be represented by a five-tuple $(b, e, m, \sigma, \alpha)$ of real numbers encoding the density

$$p_{(b,e,m,\sigma,\alpha)}(x) = \begin{cases} \alpha \cdot \mathcal{N}(m, \sigma^2) & \text{iff } b \leq x \leq e, \\ 0 & \text{otherwise,} \end{cases}$$

thus facilitating a computational representation of the corresponding mixtures as lists of such tuples. Evaluation of simple rational guards $P_{x \sim M}(x \bowtie k) \geq \theta$ over such a mixture M represented as a finite list, where k is a constant and \bowtie denotes an inequality, is also computationally feasible. Hence, a probabilistic simulator could be built provided the distributions arising in the estimate automata remain mixtures of interval-restrictions of normal distributions, which, however, would severely confine the admissible dynamics of the observed processes: non-linear dynamics, deforming the (interval-restrictions of) normal distributions, would have to be excluded, and even affine rotations are cumbersome to deal with.

An obvious alternative is the approximation of the mixtures arising as states in the estimation automata by a set of particles, as in particle-filtering [4]. As the effect of the state extrapolation within the estimation process on each of these particles can be computed whenever the state dynamics of the observed process O can be computed, due to the extrapolation concerning a single particle coinciding exactly with O 's state dynamics, simulation using such a particle approximation of the estimation mixtures is feasible even with non-linear dynamics. It should be noted, however, that extrapolation of the state of particles occurs at a different place here than in classical analysis of stochastic systems by particle-based simulation: within each state advancement step of a *single* simulation run of the overall system, we have to advance all particles representing the estimation component of the overall state-space.

Development of this technology for simulation and statistical model-checking is currently commencing in our group.

7 Summary

In the above, we have used two examples to argue that traditional hybrid-automata models, be they deterministic, nondeterministic, or stochastic, are insufficient for obtaining precise verdicts on the safety and liveness, etc., of interacting cyber-physical systems. We identified inaptness to represent rational decision-making under uncertain information as the cause of this deficiency. One may, however, argue that there might be other cures to the problem than the introduction of state estimation into the observing CPS and consequently also into its formal model, the latter necessitating a significant extension and complication of the model of hybrid automata and its related formal analysis techniques.

One such cure could be the introduction of communication within systems of interacting CPSes: the need for state estimation for an observed agent would vanish if all agents would actively communicate the local measurements that their decisions are based on (and optionally also communicate the decisions themselves) rather than having to estimate these from imprecisely observed physical behaviour. This is true, but does not provide a panacea. Not only may the mere cost as well as all kinds of reasons for data privacy and protection render such an approach undesirable, it is also bound to fail when we face socio-technical interaction within human-cyber-physical systems, as humans will neither be able nor willing to communicate a sufficiently complete representation of their perception to the CPS components. HCPSes will inherently have to rely on state estimation permitting technical systems to obtain an image of the humans' states and plans based on behavioural observations, neurophysiological measurements, etc. [8].

Another apparent cure would be the introduction of machine-learned components into the CPS for the sake of estimating (and possibly extrapolating into reasonable future) the state of observed entities. This would, however, not fundamentally change the problems induced to analysis and verification of such systems: the example of deep neural networks used for classification tasks indicates that such machine-learned state estimators manipulate “probabilities”⁴ too, necessitating a very similar treatment in models and analysis engines when trying to reason rigorously about the interactive behaviour of mutually coupled systems featuring DNN components within environmental state assessment.

We conclude that the introduction of state estimation processes, with their associated complex state spaces, is a necessary addendum to the hybrid-automata framework in order render them ripe for the modelling and analysis demands of the era of interacting intelligent systems. The development of corresponding automatic verification technology, though constituting a mostly unsolved scientific challenge, is of utmost societal importance.

References

- 1 Alessandro Abate, Joost-Pieter Katoen, John Lygeros, and Maria Prandini. Approximate model checking of stochastic hybrid systems. *Eur. J. Control*, 16(6):624–641, 2010. doi:10.3166/ejc.16.624-641.
- 2 Rajeev Alur, Costas Courcoubetis, Thomas A. Henzinger, and Pei-Hsin Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, editors, *Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*, pages 209–229. Springer, 1992. doi:10.1007/3-540-57318-6_30.
- 3 David Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012. doi:10.1017/CB09780511804779.
- 4 Karl Berntorp and Stefano Di Cairano. Particle filtering for automotive: A survey. In *22th International Conference on Information Fusion*, pages 1–8, July 2019. URL: <https://www.merl.com/publications/TR2019-069>.
- 5 L.M. Bujorianu and J. Lygeros. Toward a general theory of stochastic hybrid systems. In *Stochastic*

⁴ We are adding quotes here, as the “probability” assigned to a given label by a DNN classifier does not constitute a probability in a frequentistic sense or according to other conventional interpretations of probability theory.

- Hybrid Systems: Theory and Safety Critical Applications*, volume 337 of *LNCIS*, pages 3–30. Springer-Verlag, 2006. doi:10.1007/11587392_1.
- 6 C. Combastel. Merging Kalman filtering and zonotopic state bounding for robust fault detection under noisy environment. *IFAC-PapersOnLine*, 48(21):289–295, 2015. 9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS 2015. doi:10.1016/j.ifacol.2015.09.542.
 - 7 Christophe Coué, Cédric Pradalier, Christian Laugier, Thierry Fraichard, and Pierre Bessiere. Bayesian occupancy filtering for multitarget tracking: an automotive application. *International Journal of Robotics Research*, 25(1):19–30, January 2006. doi:10.1177/0278364906061158.
 - 8 Werner Damm, Martin Fränzle, Andreas Lüdtke, Jochem W. Rieger, Alexander Trende, and Anirudh Unni. Integrating neurophysiological sensors and driver models for safe and performant automated vehicle control in mixed traffic. In *2019 IEEE Intelligent Vehicles Symposium*, pages 82–89. IEEE, 2019. URL: <https://ieeexplore.ieee.org/xpl/conhome/8792328/proceeding>.
 - 9 M.H.A. Davis. *Markov Models and Optimization*. Chapman & Hall, London, 1993.
 - 10 J. Ding, A. Abate, and C. Tomlin. Optimal control of partially observable discrete time stochastic hybrid systems for safety specifications. In *2013 American Control Conference*, pages 6231–6236, June 2013. doi:10.1109/ACC.2013.6580815.
 - 11 Alexandre Donzé and Oded Maler. Robust satisfaction of temporal logic over real-valued signals. In Krishnendu Chatterjee and Thomas A. Henzinger, editors, *Formal Modeling and Analysis of Timed Systems - 8th International Conference, FORMATS 2010, Klosterneuburg, Austria, September 8-10, 2010. Proceedings*, volume 6246 of *Lecture Notes in Computer Science*, pages 92–106. Springer, 2010. doi:10.1007/978-3-642-15297-9_9.
 - 12 Alberto Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, June 1989. doi:10.1109/2.30720.
 - 13 Martin Fränzle, Mingshuai Chen, and Paul Kröger. In memory of Oded Maler: automatic reachability analysis of hybrid-state automata. *SIGLOG News*, 6(1):19–39, 2019. doi:10.1145/3313909.3313913.
 - 14 Martin Fränzle, Ernst Moritz Hahn, Holger Hermanns, Nicolás Wolovick, and Lijun Zhang. Measurability and safety verification for stochastic hybrid systems. In Marco Caccamo, Emilio Frazzoli, and Radu Grosu, editors, *Proceedings of the 14th ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2011, Chicago, IL, USA, April 12-14, 2011*, pages 43–52. ACM, 2011. doi:10.1145/1967701.1967710.
 - 15 Martin Fränzle, Holger Hermanns, and Tino Teige. Stochastic satisfiability modulo theory: A novel technique for the analysis of probabilistic hybrid systems. In Magnus Egerstedt and Bud Mishra, editors, *Hybrid Systems: Computation and Control, 11th International Workshop, HSCC 2008, St. Louis, MO, USA, April 22-24, 2008. Proceedings*, volume 4981 of *Lecture Notes in Computer Science*, pages 172–186. Springer, 2008. doi:10.1007/978-3-540-78929-1_13.
 - 16 Martin Fränzle and Paul Kröger. The demon, the gambler, and the engineer – reconciling hybrid-system theory with metrology. In Cliff Jones, Ji Wang, and Naijun Zhan, editors, *Symposium on Real-Time and Hybrid Systems*, volume 11180 of *Theoretical Computer Science and General Issues*, pages 165–185, Cham, 2018. Springer International Publishing. doi:10.1007/978-3-030-01461-2_9.
 - 17 Martin Fränzle and Paul Kröger. Guess what I'm doing! Rendering formal verification methods ripe for the era of interacting intelligent systems. In Tiziana Margaria and Bernhard Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation: Applications*, pages 255–272, Cham, 2020. Springer International Publishing.
 - 18 Adrian Gambier. Multivariable adaptive state-space control: A survey. In *2004 5th Asian Control Conference (IEEE Cat. No.04EX904)*, volume 1, pages 185–191 Vol.1, July 2004.
 - 19 J. Hu, J. Lygeros, and S. Sastry. Towards a theory of stochastic hybrid systems. In *Hybrid Systems: Computation and Control*, volume 1790 of *LNCIS*, pages 160–173. Springer-Verlag, 2000. doi:10.1007/3-540-46430-1_16.
 - 20 Rudolph Emil Kálmán. A new approach to linear filtering and prediction problems. *Transactions of the ASME—Journal of Basic Engineering*, 82(Series D):35–45, 1960. doi:10.1115/1.3662552.
 - 21 S. Kowalewski, M. Garavello, H. Guéguen, G. Herberich, R. Langerak, B. Piccoli, J. W. Polderman, and C. Weise. *Hybrid automata*, pages 57–86. Cambridge University Press, 2009. doi:10.1017/CB09780511807930.004.
 - 22 Helge Langseth, Thomas D. Nielsen, Rafael Rumí, and Antonio Salmerón. Inference in hybrid bayesian networks. *Reliability Engineering & System Safety*, 94(10):1499–1509, 2009. doi:10.1016/j.jress.2009.02.027.
 - 23 Eugene Lavretsky. Robust and adaptive control methods for aerial vehicles. In Kimon P. Valavanis and George J. Vachtsevanos, editors, *Handbook of Unmanned Aerial Vehicles*, pages 675–710, Dordrecht, 2015. Springer Netherlands. doi:10.1007/978-90-481-9707-1_50.
 - 24 R. P. S. Mahler. Multitarget Bayes filtering via first-order multitarget moments. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1152–1178, October 2003. doi:10.1109/TAES.2003.1261119.
 - 25 Michael Maschler, Eilon Solan, and Shmuel Zamir. *Game Theory*. Cambridge University Press, 2013. doi:10.1017/cbo9780511794216.
 - 26 Kevin P. Murphy. Switching Kalman filters, 1998.
 - 27 Kumpati S. Narendra and Zhuo Han. Adaptive control using collective information obtained from multiple models. *IFAC Proceedings Volumes*, 44(1):362–367, 2011. 18th IFAC World Congress. doi:10.3182/20110828-6-IT-1002.02237.
 - 28 Anil Nerode and Wolf Kohn. Models for hybrid systems: Automata, topologies, controllability, observability. In Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, editors,

- Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*, pages 317–356. Springer, 1992. doi:10.1007/3-540-57318-6_35.
- 29 Simo Särkkä. *Bayesian Filtering and Smoothing*. Cambridge University Press, New York, NY, USA, 2013.
- 30 C. Sherlock, A. Golightly, and C. S. Gillespie. Bayesian inference for hybrid discrete-continuous stochastic kinetic models. *Inverse Problems*, 30(11):114005, November 2014. doi:10.1088/0266-5611/30/11/114005.
- 31 Jeremy Sproston. Decidable model checking of probabilistic hybrid automata. In *Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT 2000)*, volume 1926 of *LNCS*, pages 31–45. Springer, 2000. doi:10.1007/3-540-45352-0_5.
- 32 Håkan L. S. Younes and Reid G. Simmons. Probabilistic verification of discrete event systems using acceptance sampling. In Ed Brinksma and Kim Guldstrand Larsen, editors, *Computer Aided Verification, 14th International Conference, CAV 2002, Copenhagen, Denmark, July 27-31, 2002, Proceedings*, volume 2404 of *Lecture Notes in Computer Science*, pages 223–235. Springer, 2002. doi:10.1007/3-540-45657-0_17.