Using Disparity to Enhance Test Generation for Hybrid Systems

Thao Dang and Tarik Nahhal

VERIMAG, 2 avenue de Vignate
38610 Gières, France
{Thao.Dang,Tarik.Nahhal}@imag.fr

Abstract. This paper deals with the problem of test generation for hybrid systems, which are systems with mixed discrete and continuous dynamics. In our previous work [9], we developed a coverage guided test generation algorithm, inspired by a probabilistic motion planning technique. The algorithm is guided via a process of sampling goal states, which indicate the directions to steer the system towards. In this paper, we pursue this work further and propose a method to enhance coverage quality by introducing a new notion of disparity. This notion is used to predict the situations where the goal states can not be 'directly' reached. We then develop an adaptive sampling method which permits improving coverage quality. This method was implemented and successfully applied to a number of case studies in analog and mixed-signal circuits, a domain where hybrid systems can be used as an appropriate high level model.

1 Introduction

Hybrid systems, that is, systems exhibiting both continuous and discrete dynamics, have proven to be a useful mathematical model for various physical phenomena and engineering systems. Due to the safety critical features of many such applications, much effort has been devoted to the development of automatic analysis methods and tools for hybrid systems, based on formal verification. Although these methods and tools have been successfully applied to a number of interesting case studies, their applicability is still limited to systems of small size due to the complexity of formal verification. It soon became clear that for systems of industrial size, one needs more 'light-weight' methods. Testing is another validation approach, which can be used for much larger systems and is a standard tool in industry, although it can only reveal an error but does not permit proving its absence. Although testing has been well studied in the context of finite state machines (e.g. [13] and references therein) and, more recently, of real-time systems, it has not been much investigated for continuous and hybrid systems. Therefore, a question of great interest is to bridge the gap between the verification and testing approaches, by defining a formal framework for testing of hybrid systems and developing methods and tools that help automate the testing process.

A number of special characteristics of hybrid systems make their testing particularly challenging, in particular the infiniteness of their state space and input

space. In general, in order to test an open system, one first needs to feed an input signal to the system and then check whether the behavior induced by this input signal is as expected. When there is an infinite number of possible input signals, it is important to choose the ones that lead to interesting scenarios (with respect to the property/functionality to test). Concerning existing work in test generation for hybrid systems, the paper [12] proposed a framework for generating test cases by simulating hybrid models specified using the language CHARON. The probabilistic test generation approach based on the motion planning technique **RRT** [15], proposed in [5, 11], is similar to the algorithm **gRRT** we proposed in [9]. Our method **gRRT** differs from these in the use of a coverage measure to guide the test generation. Although for various examples, the guiding tool significantly improves the coverage quality, it still suffers from a problem that we call the *controllability issue*. Roughly speaking, the guiding tool, while trying to increase the coverage, might make the system follow the directions which are unreachable. Indeed, any **RRT**-based search method which does not take into account the system's dynamics might suffer from this problem.
In this paper, we propose a method to tackle this controllability issue by introducing a new notion of disparity describing the difference between two distributions of point sets. The information of the disparity between the goal and the visited states is used to steer the exploration towards the area where the dynamics effectively allows to improve the coverage. A combination of this disparity guided method and the coverage guided algorithm **gRRT** results in a new adaptive algorithm, which we call **agRRT**. Experimental results show the coverage efficiency of the new algorithm. In terms of applications, besides traditional applications of hybrid systems (e.g. control systems), we have explored a new domain which is analog and mixed signal circuits. Indeed, hybrid systems provide a mathematical model appropriate for the modeling and analysis of these circuits. The choice of this application domain is motivated by the need in automatic tools to facilitate the design of these circuits which, for various reasons, is still lagging behind the digital circuit design.
The paper is organized as follows. We recall the hybrid systems testing framework in Section 2 and the test generation algorithm **gRRT** in Section 3, where we also discuss the controllability issue. In Section 4 we introduce the disparity notion and how to use it to enhance the coverage quality of tests. The last section is devoted to the experimental results obtained on some benchmarks of analog and mixed signal circuits.

2 Testing problem

As a model for hybrid systems, we use hybrid automata. In most classic versions of hybrid automata, continuous dynamics are defined using ordinary differential equations. However, with a view to applications to circuits where continuous dynamics are often described by differential algebraic equations, we adapt the model to capture this particularity.

A *hybrid automaton* is a tuple $\mathcal{A} = (\mathcal{X}, Q, E, F, \mathcal{I}, \mathcal{G}, \mathcal{R})$ where $\mathcal{X} \subseteq \mathbb{R}^n$ is the continuous state space; Q is a finite set of locations; E is a set of discrete transitions; $F = \{F_q \mid q \in Q\}$ such that for each $q \in Q$, $F_q = (U_q, W_q, f_q)$ defines a differential algebraic equation of the form[1]: $f_q(x(t), \dot{x}(t), u(t), w) = 0$ where the input signal $u : \mathbb{R}_+ \to U_q \subset \mathbb{R}^p$ is piecewise continuous and $w \in W_q \subset \mathbb{R}^m$ is the parameter vector; $\mathcal{I} = \{\mathcal{I}_q \subseteq \mathbb{R}^n \mid q \in Q\}$ is a set of staying conditions; $\mathcal{G} = \{\mathcal{G}_e \mid e \in E\}$ is a set of guards such that for each discrete transition $e = (q, q') \in E$, $\mathcal{G}_e \subseteq \mathcal{I}_q$; $\mathcal{R} = \{\mathcal{R}_e \mid e \in E\}$ is a set of reset maps. For each $e = (q, q') \in E$, $\mathcal{R}_e : \mathcal{G}_q \to 2^{\mathcal{I}_{q'}}$ defines how x may change when $\mathcal{A}$ switches from q to q'. The hybrid state space is $\mathcal{S} = Q \times \mathcal{X}$. A state (q, x) of $\mathcal{A}$ can change by *continuous evolution* and by *discrete evolution*. In location q, the continuous evolution of x is governed by the differential algebraic equation $f_q(x(t), \dot{x}(t), u(t), w) = 0$. Let $\phi(t, x, u(\cdot))$ be the solution of this equation with the initial condition x and under the input $u(\cdot)$. A *continuous transition* $(q, x) \overset{u(\cdot),h}{\to} (q, x')$ where $h > 0$ means that $x' = \phi(h, x, u(\cdot))$ and for all $t \in [0, h] : \phi(t, x, u(\cdot)) \in \mathcal{I}_q$. We say that $u(\cdot)$ is *admissible* starting at (q, x) for h time. For a state (q, x), if there exists a transition $e = (q, q') \in E$ and that $x \in \mathcal{G}_e$, then the transition e is enabled, the system can switch to q' and the continuous variables become $x' \in \mathcal{R}_e(x)$. This is denoted by $(q, x) \overset{e}{\to} (q', x')$, and we say that the discrete transition e is admissible at (q, x). The hybrid automata we consider are assumed to be *non-Zeno*. Note that this model is *non-deterministic* (both in continuous and discrete dynamics). To define our testing framework, we need the notions of *inputs* and *observations*.

If an input is *controllable by the tester*[2], it is called a *control input*; otherwise, it is called a *disturbance input*. A control action can be *continuous* or *discrete*. We assume that all the continuous inputs of the system are controllable. Since we want to implement the tester as a computer program, we are interested in continuous input functions that are piecewise-constant. Hence, a *continuous control action*, such as $(\bar{u}_q, h)$ specifies that the system continues with the dynamics F_q under the input $u(t) = \bar{u}_q$ for exactly h time.

For a state (q, x), a sequence of input actions $\omega = \iota_0, \iota_1, \ldots, \iota_k$ is admissible at (q, x) if: (1) ι_0 is admissible at (q, x), and (2) for each $i = 1, \ldots, k$, if (q_i, x_i) be the state such that $(q_{i-1}, x_{i-1}) \overset{\iota_{i-1}}{\to} (q_i, x_i)$, then ι_i is admissible at (q_i, x_i). The sequence $(q, x), (q_1, x_1) \ldots, (q_k, x_k)$ is called the *trace* starting at (q, x) under ω and is denoted by $\tau((q, x), \omega)$.

Since the tester cannot manipulate uncontrollable actions, we need the notion of admissible control action sequences. However, due to space limitation, we do not include its formal definition, which can be found in [3]. Intuitively, if we apply a control action sequence to the automaton, some disturbance actions can occur between the control actions. A control action sequence that does not cause the automaton to be blocked is called admissble. The set of traces starting at (q, x) after an admissible control action sequence ω is denoted by $Tr((q, x), \omega)$. We

[1] We assume the existence and uniqueness of solutions of these equations.

[2] By 'controllable' here we mean that the tester can manipulate this input, and it should not be confused with the term 'controllable' in control theory.

denote by $S_{\mathcal{C}}(\mathcal{A})$ the *set of all admissible control action sequences* starting at an initial state (q_{init}, x_{init}).
We assume that the location of the hybrid automaton $\mathcal{A}$ is observable by the tester. We also assume a set $V_o(\mathcal{A})$ of observable continuous variables of $\mathcal{A}$. The projection of a continuous state x of $\mathcal{A}$ on $V_o(\mathcal{A})$, denoted by $\pi(x, V_o(\mathcal{A}))$, is called an *observation*. The projection can be then defined for a trace. Let ω be an admissible control action sequence starting at an initial state (q_{init}, x_{init}) of $\mathcal{A}$. The set of *observation sequences* associated with ω is $S_{\mathcal{O}}(\mathcal{A}, \omega) = \{\pi(\tau, V_o(\mathcal{A})) \mid \tau \in Tr((q_{init}, x_{init}), \omega)\}$.
In our framework, the specification is modeled by a hybrid automaton $\mathcal{A}$ and the system under test SUT (e.g. an implementation) by another hybrid automaton $\mathcal{A}_s$ such that $V_o(\mathcal{A}) \subseteq V_o(\mathcal{A}_s)$ and $S_{\mathcal{C}}(\mathcal{A}) \subseteq S_{\mathcal{C}}(\mathcal{A}_s)$ (that is, a control sequence which is admissible for $\mathcal{A}$ is also admissible for $\mathcal{A}_s$). Note that we do not assume that we know the model $\mathcal{A}_s$. The goal of testing is to make statements about the relation between the traces of the SUT and those of the specification. The tester performs experiments on $\mathcal{A}_s$ in order to study the relation between $\mathcal{A}$ and $\mathcal{A}_s$. It emits an admissible control sequence to the SUT and measures the resulting observation sequence in order to produce a verdict ('pass', or ' fail', or 'inconclusive'). The observations are measured at the end of each continuous control action and after each discrete (disturbance or control) action.

Definition 1 (Conformance). *The system under test $\mathcal{A}_s$ is conform to the specification $\mathcal{A}$, denoted by $\mathcal{A} \approx \mathcal{A}_s$, iff* $\forall \omega \in S_{\mathcal{C}}(\mathcal{A}) : \ \pi(S_{\mathcal{O}}(\mathcal{A}_s, \omega), V_o(\mathcal{A})) \subseteq S_{\mathcal{O}}(\mathcal{A}, \omega)$.

A *test case* is represented by a (finite) tree where each node is associated with an observation and each edge with a control action. A hybrid automaton might have an infinite number of infinite traces; however, the tester can only perform a finite number of test cases in finite time. Therefore, we need to select a finite portion of the input space of $\mathcal{A}$ and test the conformance of $\mathcal{A}_s$ with respect to this portion. The selection is done using a *coverage criterion* that we formally define in the following.

Star discrepancy coverage. We are interested in defining a coverage measure that describes how 'well' the visited states represent the reachable set. This measure is defined using the *star discrepancy*, which is an important notion in equidistribution theory as well as in quasi-Monte Carlo techniques [1].
We first define the coverage for each location. Since a hybrid system can only evolve within the staying sets of the locations, we are interested in the coverage over these sets. For simplicity we assume that all the staying sets are boxes. If a staying set $\mathcal{I}_q$ is not a box, we can take the smallest oriented box that encloses $\mathcal{I}_q$, and apply the star discrepancy definition to the oriented box after an appropriate coordination change. Let P be a set of k points inside $\mathcal{B} = [l_1, L_1] \times \ldots \times [l_n, L_n]$, which is the staying set of the location q. Let Γ be the set of all sub-boxes J of the form $J = \prod_{i=1}^{n} [l_i, \beta_i]$ with $\beta_i \in [l_i, L_i]$ (see Figure 1 for an illustration). The local discrepancy of the point set P with respect to

the subbox J is $D(P,J) = |\frac{A(P,J)}{k} - \frac{\lambda(J)}{\lambda(\mathcal{B})}|$ where $A(P,J)$ is the number of points of P inside J, and $\lambda(J)$ is the volume of J. The star discrepancy of P with respect to the box $\mathcal{B}$ is defined as: $D^*(P,\mathcal{B}) = sup_{J\in\Gamma} D(P,J)$. Note that $0 < D^*(P,\mathcal{B}) \leq 1$.

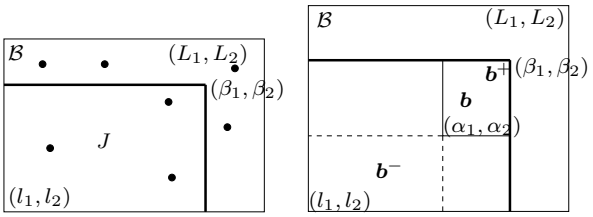

Fig. 1. Illustration of the star discrepancy notion.

Intuitively, the star discrepancy is a measure for the irregularity of a set of points. A large value $D^*(P,\mathcal{B})$ means that the points in P are not well equidistributed over $\mathcal{B}$. When the region is a hyper-cube, the star discrepancy measures how badly the point set estimates the volume of the cube.
Let $\mathcal{P} = \{(q, P_q) \mid q \in Q \ \wedge \ P_q \subset \mathcal{I}_q\}$ be the set of hybrid states. We define the coverage for each location $q \in Q$ as $Cov_q(\mathcal{P}) = 1 - D^*(P_q, \mathcal{I}_q)$.

Definition 2 (Star discrepancy coverage). *The coverage of $\mathcal{P}$ is defined as: $Cov(\mathcal{P}) = \frac{1}{||Q||} \sum_{q\in Q} Cov_q(\mathcal{P})$ where $||Q||$ is the number of locations in Q.*

3 Coverage guided test generation

In this section we recall the **gRRT** algorithm [9], which is a combination of the Rapidly-exploring Random Tree (**RRT**) algorithm, a successful robot motion planning technique (see for example [15]), and a guiding tool used to achieve a good test coverage. Essentially, the algorithm constructs a tree $\mathcal{T}$ as follows (see Algorithm 1). First, from the set of initial states, we sample a finite number of initial states, each corresponds to a root of the tree. Similarly, we can consider a finite number of parameter values and associate them with each initial state. Along a path from each root, the parameter vector remains constant.
In each iteration, a goal state s_{goal} is sampled. A neighbor $s_{near} = (q_{near}, x_{near})$ of s_{goal} is then determined. This neighbor is used as the starting state for the current iteration. To define a neighbor of a state, we define a hybrid distance from $s_1 = (q_1, x_1)$ to $s_2 = (q_2, x_2)$ as an average length of all the potential traces from s_1 to s_2 (see [9] for more detail). In ContinuousStep, we want to find an input $\bar{u}_{q_{near}}$ to take the system from s_{near} towards s_{goal} as closely as possible after h time, which results in a new state s_{new}. To find $\bar{u}_{q_{near}}$, when the set U is not finite it can be sampled, or one can solve a local optimal control problem. Then, from s_{new}, we compute its successors by all possible discrete transitions.

Note that the simulator can also detect the uncontrollable discrete transitions that become enabled during a continuous step, and in this case the current continuous step is stopped (which is equivalent to using a variable time step). The algorithm terminates after some maximal number of iterations. To extract a test case from the tree, we project the states at the nodes on the observable variables of $\mathcal{A}$.

In the classic **RRT** algorithms, which work in a continuous setting, only x_{goal} needs to be sampled, and a commonly used sampling distribution of x_{goal} is uniform over the state space. In addition, the point x_{near} is defined as a nearest neighbor of x_{goal} in some usual distance, such as the Euclidian distance. In our **gRRT** algorithm, the goal state sampling is not uniform and the function GuidedSampling plays the role of guiding the exploration by sampling of goal states according to the current coverage of the visited states.

Algorithm 1 Test generation algorithm **gRRT**

$\mathcal{T}.init(s_{init})$, $j = 1$ ▷ s_{init}: initial state
repeat
  $s_{goal} = \textsc{GuidedSampling}(\mathcal{S})$ ▷ $\mathcal{S}$: hybrid state space
  $s_{near} = \textsc{Neighbor}(\mathcal{T}, s_{goal})$
  $(s_{new}, \bar{u}_{q_{near}}) = \textsc{ContinuousStep}(s_{near}, h)$
  $\textsc{DiscreteSteps}(\mathcal{T}, s_{new})$, $j++$
until $j \geq J_{max}$

Coverage guided sampling. To evaluate the coverage of a set of states, we estimate a lower and upper bound of the star discrepancy (exact computation is well-known to be a hard problem). These bounds as well as the information obtained from their estimation are used to decide which parts of the state space have been 'well explored' and which parts need to be explored more. Let us briefly describe this estimation method (see [9]). Let $\mathcal{B} = [l_1, L_1] \times \ldots \times [l_n, L_n]$. We define a box partition of $\mathcal{B}$ as a set of boxes $\Pi = \{\boldsymbol{b}^1, \ldots, \boldsymbol{b}^m\}$ such that $\cup_{i=1}^{m} \boldsymbol{b}^i = \mathcal{B}$ and the interiors of the boxes $\boldsymbol{b}^i$ do not intersect. Each such box is called an *elementary box*. Given a box $\boldsymbol{b} = [\alpha_1, \beta_1] \times \ldots \times [\alpha_n, \beta_n] \in \Pi$, we define $\boldsymbol{b}^+ = [l_1, \beta_1] \times \ldots \times [l_n, \beta_n]$ and $\boldsymbol{b}^- = [l_1, \alpha_1] \times \ldots \times [l_n, \alpha_n]$ (see Figure 1). For any finite box partition Π of $\mathcal{B}$, an upper bound $B(P, \Pi)$ and a lower bound $C(P, \Pi)$ of the star discrepancy $D^*(P, \mathcal{B})$ can be written as: $B(P, \Pi) = \max_{\boldsymbol{b} \in \Pi} \max\{\frac{A(P, \boldsymbol{b}^+)}{k} - \frac{\lambda(\boldsymbol{b}^-)}{\lambda(\mathcal{B})}, \frac{\lambda(\boldsymbol{b}^+)}{\lambda(\mathcal{B})} - \frac{A(P, \boldsymbol{b}^-)}{k}\}$ and $C(P, \Pi) = \max_{\boldsymbol{b} \in \Pi} \max\{|\frac{A(P, \boldsymbol{b}^-)}{k} - \frac{\lambda(\boldsymbol{b}^-)}{\lambda(\mathcal{B})}|, |\frac{A(P, \boldsymbol{b}^+)}{k} - \frac{\lambda(\boldsymbol{b}^+)}{\lambda(\mathcal{B})}|\}$.

To sample a goal state, we first sample a discrete location and then a continuous state. Let $\mathcal{P} = \{(q, P_q) \mid q \in Q \wedge P_q \subset \mathcal{I}_q\}$ be the current set of visited states. The discrete location sampling distribution depends on the current coverage of each location: $Pr[q_{goal} = q] = \frac{D^*(P_q, \mathcal{I}_q)}{\sum_{q' \in Q} D^*(P_{q'}, \mathcal{I}_{q'})}$. To sample x_{goal}, we first sample an elementary box $\boldsymbol{b}_{goal}$ from the set Π, then we sample a point x_{goal}

in $\boldsymbol{b}_{goal}$ uniformly. The elementary box sampling distribution is biased in order to improve the coverage. We favor the selection of an elementary box such that a new point x added in this box results in a reduction of the lower and upper bounds(see [9]).
To demonstrate the performance of **gRRT**, we use two illustrative examples. For brevity, we call the classical **RRT** algorithm using uniform sampling and the Euclidian metric **hRRT**. The reason we choose these examples is that they differ in the reachability property. In the first example, the system is 'controllable' in the sense that the whole state space is reachable from the initial states (by using appropriate inputs), but in the second example the reachable set is only a small part of the state space.

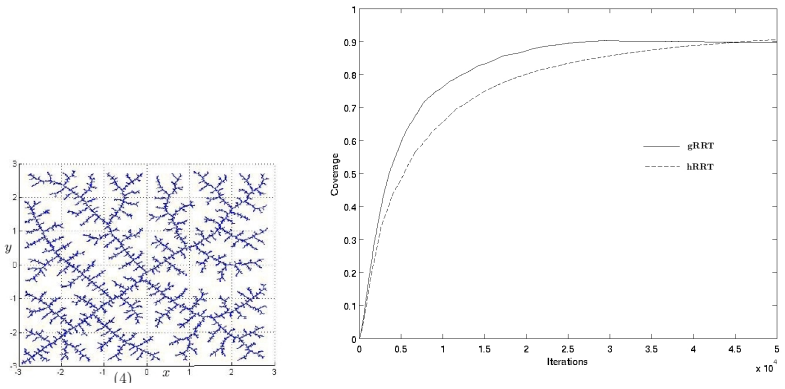

Fig. 2. Left: The **gRRT** exploration result. Right: Test coverage evolution.

Example 1. This is a two-dimensional continuous system where the state space $\mathcal{X}$ is a box $\mathcal{B} = [-3, 3] \times [-3, 3]$. The continuous dynamics is $f(x, t) = u(t)$ where the input set is $U = \{u \in \mathbb{R}^2 \mid ||u|| \leq 0.2\}$.

We use 100 input values resulting from a discretization of the set U. The initial state is $(-2.9, -2.9)$. The time step is 0.002. Figure 2 shows the result obtained using **gRRT** and the evolution of the coverage of the states generated by **gRRT** (solid curve) and by **hRRT** (dashed curve). The figure indicates that **gRRT** achieved a better coverage quality, especially in convergence rate.

Example 2. This example is a linear system with a stable focus at the origin. Its dynamics is as follows: $\dot{x} = -x - 1.9y + u_1$ and $\dot{y} = 1.9x - y + u_2$. We let the dynamics be slightly perturbed by an additive input u. The state space is the box $\mathcal{B} = [-3, 3] \times [-3, 3]$. The input set $U = \{u \in \mathbb{R}^2 \mid ||u|| \leq 0.2\}$. Figure 3 shows the results obtained after 50000 iterations. We can see that again the guided sampling method achieved a better coverage result.

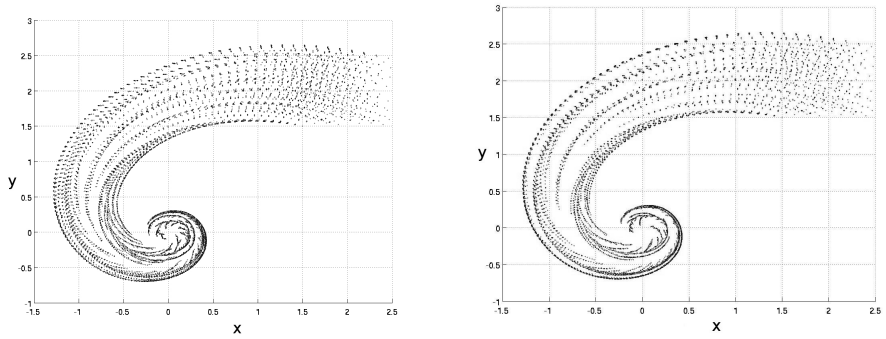

Fig. 3. Results obtained using the guided sampling method (left) and using the uniform sampling method (right).

Controllability issue. From different experiments with Example 2, we observed that the coverage performance of **gRRT** is not satisfying when the reachable space is only a small part of the whole state space. This can be explained as follows. There are boxes, such as those near the bottom right vertex of the bounding box, which have a high potential of reducing the bounds of the star discrepancy. Thus, the sampler frequently selects these boxes. However, these boxes are not reachable from the initial states, and all attempts to reach them do not expand the tree beyond the boundary of the reachable set. This results in a large number of points concentrated near this part of the boundary, while other parts of the reachable set are not well explored.
It is important to emphasize that this problem is not specific to **gRRT**. The **RRT** algorithm using the uniform sampling method and, more generally, any algorithm that does not take into account the differential contraints of the system, may suffer from this phenomenon. This phenomenon can however be captured by the evolution of the disparity between the set of goal states and the set of visited states. This notion will be formally defined in the next section. Roughly speaking, it describes how different their distributions are. When the disparity does not decrease after a certain number of iterations, this often indicates that the system cannot approach the goal states, and it is better not to favor an expansion towards the exterior but a *refinement*, that is an exploration in the interior of the already visited regions.
Figure 4 shows the evolution of the disparity between the set P^k of visited states at the k^{th} iteration and the set G^k of goal states for the two examples. We observe that for the system of Example 1 which can reach any state in the state space, the visited states follow the goal states, and thus the disparity gets stabilized over time. However, in Example 2, where the system cannot reach everywhere, the disparity does not decrease for a long period of time, during which most of the goal states indicate unreachable directions.

Figure 4 shows the Voronoi diagram of a set of visited states. The boundary of the reachable set can be seen as an 'obstacle' that prevents the system from crossing it. Note that the Voronoi cells of the states on the boundary are large (because they are near the large unvisited part of the state space). Hence, if the goal states are uniformly sampled over the whole state space, these large Voronoi cells have higher probabilities of containing the goal states, and thus the exploration is 'stuck' near the boundary, while the interior of the reachable set is not well explored.

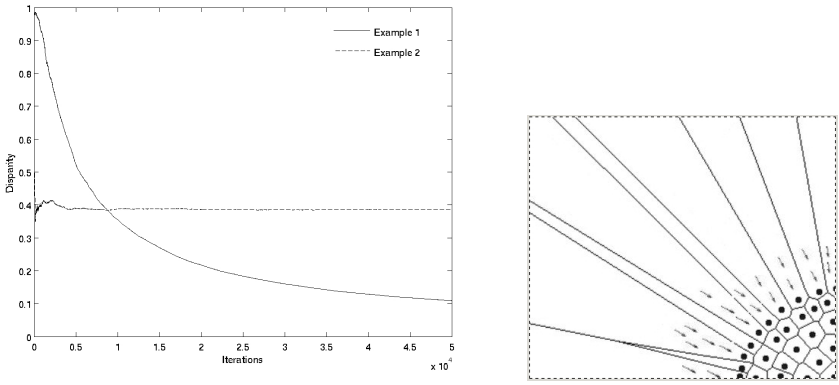

Fig. 4. Left: Disparity between the visited states and the goal states.

To tackle this problem, we introduce the notion of disparity to describe the 'difference' in the distributions of two sets of points. The controllability problem can be detected by a large value of the disparity between the goal states and the visited states. We can thus combine **gRRT** with a disparity based sampling method, in order to better adapt to the dynamics of the system. This is the topic of the next section.

4 Disparity guided sampling

The notion of disparity between two point sets that we develop here is inspired by the star discrepancy. Indeed, by definition, the star discrepancy of a set P w.r.t. the box $\mathcal{B}$ can be seen as a comparison between P and an 'ideal' infinite set of points distributed all over $\mathcal{B}$. Let P and Q be two sets of points inside $\mathcal{B}$. Let J be a sub-box of $\mathcal{B}$ which has the same bottom-left vertex as $\mathcal{B}$ and the top-right vertex of which is a point inside $\mathcal{B}$. Let Γ be the set of all such sub-boxes. We define the local disparity between P and Q with respect to the sub-box J as: $\gamma(P,Q,J) = |\frac{A(P,J)}{||P||} - \frac{A(Q,J)}{||Q||}|$ where $A(P,J)$ is the number of points of P inside J and $||P||$ is the total number of points of P.

Definition 3 (Disparity). *The disparity between P and Q with respect to the bounding box $\mathcal{B}$ is defined as: $\gamma^*(P,Q,\mathcal{B}) = sup_{J\in\Gamma}\gamma(P,Q,J)$.*

The disparity satisfies $0 < \gamma^*(P,Q,\mathcal{B}) \leq 1$. A small value $\gamma^*(P,Q,\mathcal{B})$ means that the distributions of the sets P and Q over the box $\mathcal{B}$ are 'similar'. To illustrate our notion of disparity, we consider two well-known sequences of points: the Faure and Halton sequences [6, 14], shown in Figure 5. Their disparity is 0.06, indicating that they have similar distributions. The second example is used to compare the Faure sequence and a set of 100 points concentratred in some small rectangle, and the disparity between them is large (0.54).
The exact computation of the disparity is as hard as the exact computation of the star discrepancy, which is due to the infinite number of the sub-boxes. We propose a method for estimating a lower and an upper bound for this new measure. Let Π be a box partition of $\mathcal{B}$. Let P, Q be two sets of points inside $\mathcal{B}$. For each elementary box $\boldsymbol{b} \in \Pi$ we denote $\mu_m(\boldsymbol{b}) = \max\{\mu_c(\boldsymbol{b}), \mu_o(\boldsymbol{b})\}$ where $\mu_c(\boldsymbol{b}) = \dfrac{A(P,\boldsymbol{b}^+)}{||P||} - \dfrac{A(Q,\boldsymbol{b}^-)}{||Q||}$, $\mu_o(\boldsymbol{b}) = \dfrac{A(Q,\boldsymbol{b}^+)}{||Q||} - \dfrac{A(P,\boldsymbol{b}^-)}{||P||}$. We also denote $c(\boldsymbol{b}) = \max\{|\dfrac{A(P,\boldsymbol{b}^-)}{||P||} - \dfrac{A(Q,\boldsymbol{b}^-)}{||Q||}|, |\dfrac{A(P,\boldsymbol{b}^+)}{||P||} - \dfrac{A(Q,\boldsymbol{b}^+)}{||Q||}|\}$.

Theorem 1. [Upper and lower bounds] *An upper bound $B_d(P,Q,\Pi)$ and a lower bound $C_d(P,Q,\Pi)$ of the disparity between P and Q are: $B_d(P,Q,\Pi) = \max_{\boldsymbol{b}\in\Pi}\{\mu_m(\boldsymbol{b})\}$ and $C_d(P,Q,\Pi) = \max_{\boldsymbol{b}\in\Pi}\{c(\boldsymbol{b})\}$.*

For each elementary box $\boldsymbol{b} = [\alpha_1,\beta_1]\times\ldots\times[\alpha_n,\beta_n] \in \Pi$, we define the $\mathcal{W}$-zone, denoted by $\mathcal{W}(\boldsymbol{b})$, as follows: $\mathcal{W}(\boldsymbol{b}) = \boldsymbol{b}^+ \setminus \boldsymbol{b}^-$.

Theorem 2. *A bound on the disparity estimation error is:*
$B_d(P,Q,\Pi) - C_d(P,Q,\Pi) \leq \max\limits_{\boldsymbol{b}\in\Pi} \max\{\dfrac{A(P,\mathcal{W}(\boldsymbol{b}))}{||P||}, \dfrac{A(Q,\mathcal{W}(\boldsymbol{b}))}{||Q||}\}$.

The above error bounds can be used to dynamically refine the partition. The proofs of the above results can be found in [3].

Disparity guided sampling. The essential idea of our disparity based sampling method is to detect when the dynamics of the system does not allow the tree to expand towards the goal states and then to avoid such situations by favoring a refinement, that is an exploration near the already visited states.
A simple way to bias the sampling towards the set P^k of already visited states is to reduce the sampling space. Indeed, we can make a bounding box of the set P^k and give more probability of sampling inside this box than outside it. Alternatively, we can guide the sampling using the disparity information as follows. The objective now is to reduce the disparity between the set G^k of goal states and the set P^k of visited states. Like the guiding method using the star discrepancy, we define for each elementary box $\boldsymbol{b}$ of the partition a function $\eta(\boldsymbol{b})$ reflecting the potential for reduction of the lower and upper bounds of the disparity between

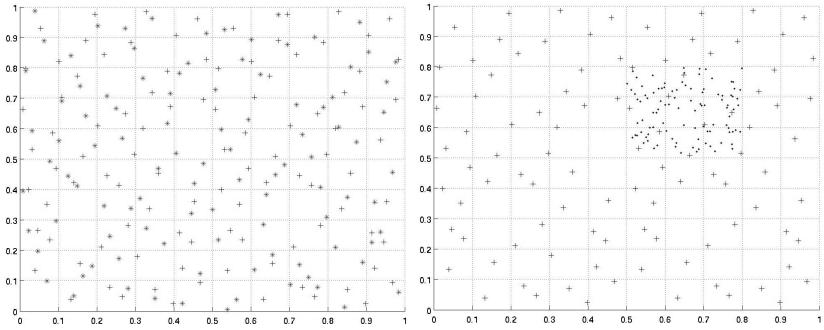

Fig. 5. Left: Faure sequence (+ signs) and Halton sequence (∗ signs). Right: Faure sequence (+ signs) and another C pseudo-random sequence (∗ signs).

P^k and G^k. This means that we favor the selection of the boxes that make the distribution of goal states G^k approach that of the visited states P^k. Choosing such boxes can improve the quality of refinement. The formulation of the potential influence function for the disparity-based sampling method is similar to that for the coverage guided sampling.

A combination of the coverage guided and the disparity guided sampling methods is done as follows. We fix a time window T_s and a threshold ϵ. When using the coverage guided method, if the algorithm detects that the disparity between the G^k and P^k does not decrease by ϵ after T_s time, it switches to the disparity guided method until the disparity is reduced more significantly and switches back to the coverage guided method. Note that a non-decreasing evolution of the disparity is an indication of the inability of the system to approach the goal states. In an interactive exploration mode, it is possible to let the user to manually decide when to switch. As mentioned earlier, we call the resulting algorithm **agRRT** (the letter 'a' in this acronym stands for 'adaptive').

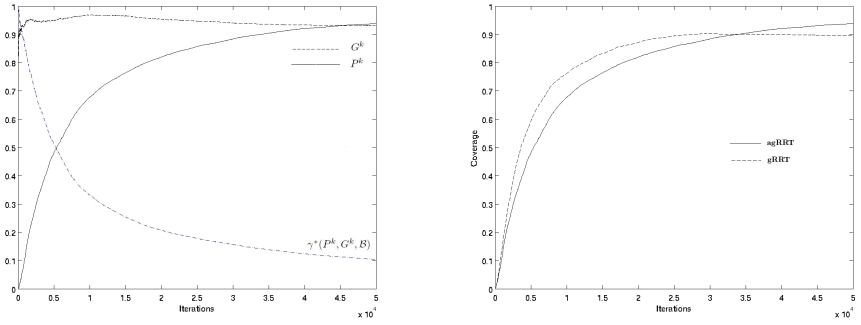

Fig. 6. Left: Test coverage of the result obtained using **agRRT** for Example 1.

We use the examples in the previous section to demonstrate the coverage improvement of **agRRT**. Figure 6 shows that the final result for Example 1 obtained using **agRRT** has a better coverage than that obtained using **gRRT**. The solid curve represents the coverage of the set P^k of visited states and the dashed one the coverage of the set G^k of goal states. The dash-dot curve represents the disparity between G^k and P^k. The result obtained using **agRRT** for Example 2 is shown in Figure 7, which also indicates an improvement in coverage quality. The figure on the right shows the set of generated goal states, drawn in dark color. In this example, we can observe the adaptivity of the combination of **gRRT** and **agRRT**. Indeed, in the beginning, the **gRRT** algorithm was used to rapidly expand the tree. After some time, the goal states sampled from the outside of the exact reachable space do not improve the coverage, since they only create more states near the boundary of the reachable set. In this case, the disparity between P^k and G^k does not decrease, and the **agRRT** is thus used to enable an exploration in the interior of the reachable set. The interior the reachable set thus has a higher density of sampled goal states than the outside, as one can see in the figure.

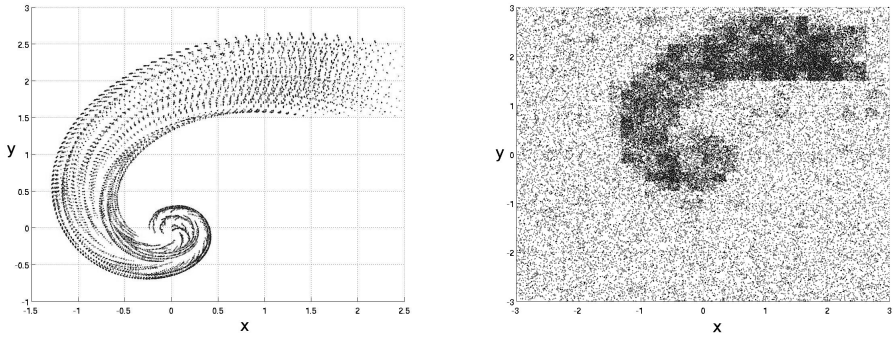

Fig. 7. Result obtained using **agRRT**. Left: visited states P^k, right: goal states G^k.

5 Applications to analog and mixed-signal circuits

Using the above results, we implemented a test generation tool and tested it on a number of control applications, which proved its scalability to high dimensional systems [9]. In this implementation, all the sets encountered in the hybrid automaton definition are convex polyhedra. For circuit applications, we use the well-known RADAU algorithm for solving differential algebraic equations (DAE). We recall that solving high index and stiff DAEs is computationally expensive, and in order to evaluate the efficiency of the test generation algorithm, we have chosen two practical circuits with DAEs of this type. The three circuits we treated are: a transistor amplifier, a voltage controlled oscillator, and a

Delta-Sigma modulator circuit. As described earlier, we use a criterion based on the disparity to automatically detect the controllability problem and to switch to the disparity guided strategy for some time. This criterion is necessary because it is not easy to anticipate whether the system under study is 'highly controllable' or not. However, prior knowledge of the system's behavior can be helpful in the decision of favoring one strategy over the other. Among the circuit benchmarks we treated, the Delta-Sigma circuit has an expansive dynamics and thus we could predict that the coverage-guided strategy is appropriate. However, for the transistor amplifier and the voltage controlled oscillator circuit, a more frequent use of the disparity guided strategy was necessary.

Transistor amplifier. The first example is a transistor amplifier model [4], shown in Figure 8, where the variable y_i is the voltage at node i; U_e is the input signal and $y_8 = U_8$ is the output voltage. The circuit equations are a system of non-linear DAEs of index 1 with 8 continuous variables.The circuit parameters are: $U_b = 6$; $U_F = 0.026$; $R_0 = 1000$; $R_k = 9000$, $k = 1, \ldots, 9$; $C_k = k10^{-6}$; $\alpha = 0.99$; $\beta = 10^{-6}$. The initial state $y_{init} = (0, U_b/(R_2/R_1 + 1), U_b/(R_2/R_1 + 1), U_b, U_b/(R_6/R_5 + 1), U_b/(R_6/R_5 + 1), U_b, 0)$. To study the

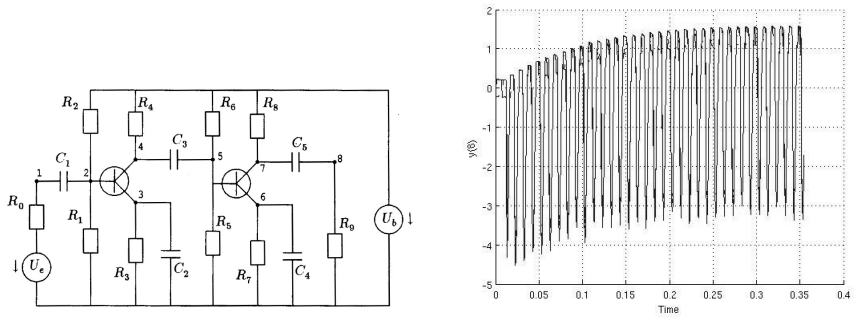

Fig. 8. Test generation result for the transistor amplifier.

influence of circuit parameter uncertainty, we consider is a perturbation in the relation between the current through the source of the two transistors and the voltages at the gate and source $I_S = g(U_G - U_S) = \beta(e^{\frac{U_G - U_S}{U_F}} - 1) + \epsilon$, with $\epsilon \in [\epsilon_{min}, \epsilon_{max}] = [-5e-5, 5e-5]$. The input signal $U_e(t) = 0.1sin(200\pi t)$. The acceptable interval of U_8 in the non-perturbed circuit is $[-3.01, 1.42]$. Once the initial transient period has settled down, the generated test case leads to an overshoot after 18222 iterations (corresponding to 1.1mn of computation time). The total computation time for generating 50000 states was 3mn. Figure 8 shows the generated states projected on U_8 over the first 0.03 seconds.

Voltage controlled oscillator. The circuit [7] is described by a system of DAEs with 55 continuous variables. The oscillating frequency of the variables v_{C_1} and v_{C_2} is a linear function of the input voltage u_{in}. We study the influence of

a time-variant perturbation in C_2, modeled as an input signal, on this frequency. We show that, in addition to conformance relation, using this framework, we can test a property of the input/output relation. The oscillating period $T \pm \delta$ of v_{C_2} can be expressed using a simple automaton with one clock y in Figure 9. The question is to know if given an oscillating trace in $\mathcal{A}$, its corresponding trace in $\mathcal{A}_s$ is also oscillates with the same period. This additional automaton can be used to determine test verdicts for the traces in the computed test cases. If an observation sequence corresponds to a path entering the 'Error' location, then it causes a 'fail' verdict. Since we cannot use finite traces to prove a safety property, the set of obsevation sequences that cause a 'pass' verdict is empty, and therefore the remaining obsevation sequences (that do not cause a 'fail' verdict) cause a 'inconclusive' verdict. We consider a constant input voltage

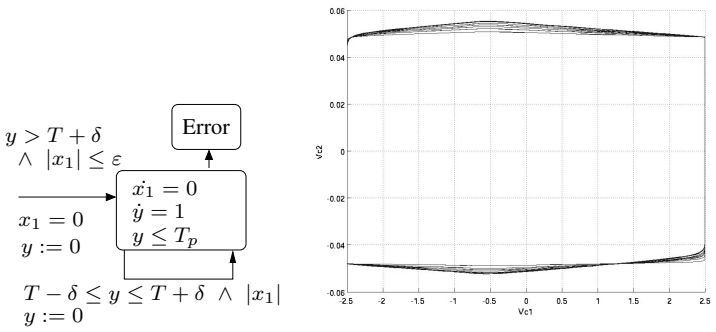

Fig. 9. Left: Automaton for an oscillation specification. Right: Variable v_{C_2} over time.

$u_{in} = 1.7$. The coverage measure was defined on the projection of the state space on v_{C_1} and v_{C_2}. The generated test case shows that *after the transient time*, under a time-variant deviation of C_2 which ranges within $\pm 10\%$ of the value of $C_2 = 0.1e-4$, the variables v_{C_1} and v_{C_2} oscillate with the period $T \in [1.25, 1.258]s$ (with $\varepsilon = 2.8e-4$). This result is consistent with the result presented in [7]. The number of generated states was 30000 and the computation time was 14mn. Figure 9 shows the explored traces of v_{C_2} over time.

Delta-Sigma circuit. We consider a third-order Delta-Sigma modulator [10], which is a mixed-signal circuit. When the input sinusoid is positive and its value is less than 1, the output takes the $+1$ value more often and the quantization error is fed back with negative gain and 'accumulated' in the integrator $\frac{1}{z-1}$. Then, when the accumulated error reaches a certain threshold, the quantizer switches the value of the output to -1 to reduce the mean of the quantization error. A third-order Delta-Sigma modulator is modeled as a hybrid automaton, shown in Figure 10. The discrete-time dynamics of the system is as follows: $x(k+1) = Ax(k) + bu(k) - sign(y(k))a$, $y(k) = c_3x_3(k) + b_4u(k)$ where $x(k) \in \mathbb{R}^3$ is the integrator states, $u(k) \in \mathbb{R}$ is the input, $y(k) \in \mathbb{R}$ is the input of the quantizer. Thus, its output is $v(k) = sign(y(k))$, and one can see that whenever v remains

constant, the system dynamics is affine continuous. A modulator is stable if under a bounded input, the states of its integrators are bounded. The test generation

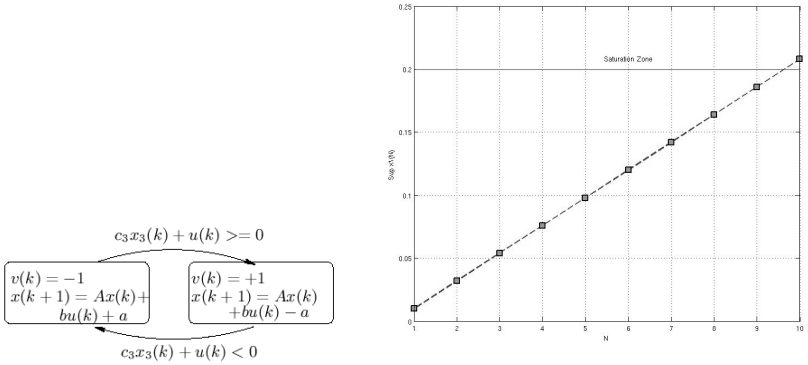

Fig. 10. Model of a third-order modulator and test generation result.

algorithm was performed for the initial state $x(0) \in [-0.01, 0.01]^3$ and the input values $u(k) \in [-0.5, 0.5]$. After exploring only 57 states, saturation was already detected. The computation time was less than 1 second. Figure 10 shows the values of $(\sup x_1(k))_k$ as a function of the number k of time steps. We can see that the sequence $(\sup x_1(k))_k$ leaves the safe interval $[-x_1^{sat}, x_1^{sat}] = [-0.2, 0.2]$, which indicates the instability of the circuit. This instability for a fixed finite horizon was also detected in [2] using an optimization-based method.

6 Related work and conclusions

In this paper, we described the **agRRT** algorithm which is a combination of the coverage guided test generation algorithm **gRRT** and a disparity guided algorithm. The latter uses the information about the disparity between the goal states and the visited states in order to steer the exploration towards the area where the dynamics of the system allows to better improve the test coverage. We provided some examples to show the efficiency of this guiding tool, in terms of coverage improvement. We also reported some experimental results where our test generation tool successfully treated a number of benchmarks in analog and mixed-signal circuits.

Concerning related work along this line, sampling the configuration space has been a fundamental problem in probabilistic motion planning. Our idea of guiding the simulation via the sampling process has some similarity with the sampling domain control [15]. In this work, the domains over which the goal points are sampled need to reflect the geometric and differential constraints of the system, and more generally, the controllability of the system. In [8], another method for biasing the exploration was proposed. The main idea of this method is to reduce the dispersion in an incremental manner. This idea is thus very close to the idea

of our guiding method in spirit; however, their concrete realizations are different. This method tries to lower the dispersion by using K samples in each iteration (instead of a single sample) and then select from them a best sample by taking into account the feasibility of growing the tree towards it. Finally, we mention that the controllability issue was addressed in [5] where the number of successful iterations is used to define an adaptive biased sampling.
A number of directions for future research can be identified. First, we are interested in enriching our framework to capture partial observability and measurement imprecisions. To facilitate the application to practical circuits, we need a tool for automatic generation of hybrid automata from commonly-used circuit descriptions, such as SPICE netlists.

Acknowledgements. We thank E. Frazzoli, O. Maler, and S. LaValle for their valuable comments. This work is supported by the project ANR VAL-AMS.

References

1. J. Beck and W. W. L. Chen. Irregularities of distribution. In *Acta Arithmetica*, Cambridge University Press, 1997.
2. T. Dang, A. Donze and O. Maler. Verification of analog and mixed-signal circuits using hybrid systems techniques. FMCAD'04, LNCS 3312, Springer, 2004.
3. T. Dang and T. Nahhal, Coverage-Guided Test Generation for Continuous and Hybrid Systems. Technical report, Verimag, Grenoble, 2007.
4. C. Lubich, E. Hairer and M. Roche. The numerical solution of differential-algebraic systems by Runge Kutta methods. *Lecture Notes in Math. 1409*, Springer, 1989.
5. J. Esposito, J. W. Kim, and V. Kumar. Adaptive RRTs for validating hybrid robotic control systems. *Workshop on Algorithmic Foundations of Robotics*, 2004.
6. Henri Faure. Discrépance de suites associées à un système de numération. *Acta Arithmetica*, 41:337–351, 1982.
7. D. Grabowski, D. Platte, L. Hedrich, and E. Barke. Time constrained verification of analog circuits using model-checking algorithms. *Electr. Notes Theor. Comput. Sci.*, 153(3):37–52, 2006.
8. S. R. Lindemann and S. M. LaValle. Incrementally reducing dispersion by increasing Voronoi bias in RRTs. *IEEE Conf. on Robotics and Automation*, 2004.
9. T. Nahhal and T. Dang. Test coverage for continuous and hybrid systems. *Computer Aided Verification* CAV'07, pages 454–468, 2007.
10. B. Pérez-Verdú, F. Medeiro, and A. Rodríguez-Vázquez. Top-Down Design of High-Performance Sigma-Delta Modulators, Kluwer Academic Publishers, 2001.
11. E. Plaku, L. E. Kavraki, and M. Y. Vardi. Hybrid systems: From verification to falsification. *Computer Aided Verification* CAV'07, LNCS 4590, Springer, 2007.
12. L. Tan, J. Kim, O. Sokolsky, and I. Lee. Model-based testing and monitoring for hybrid embedded systems. *Information Reuse and Integration* IRI'04, 2004.
13. Jan Tretmans. Testing concurrent systems: A formal approach. *10th Int. Conf. on Concurrency Theory* CONCUR '99, pages 46–65, Springer, 1999.
14. X. Wang and F. Hickernell. Randomized Halton sequences, Math. Comp. Modelling, 32:887–899, 2000.
15. A. Yershova, L. Jaillet, T. Simeon, and S. LaValle. Dynamic-domain rrts: Efficient exploration by controlling the sampling domain. *IEEE Int. Conf. Robotics and Automation*, 2005.