Enhancing sampling-based kinodynamic motion planning for quadrotors

Conference Paper · October 2015

DOI: 10.1109/IROS.2015.7353709

Alexandre Boeuf
Donecle
7 PUBLICATIONS **41** CITATIONS
SEE PROFILE

Juan Cortés
French National Centre for Scientific Research
115 PUBLICATIONS **3,422** CITATIONS
SEE PROFILE

Rachid Alami
Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS)
316 PUBLICATIONS **9,708** CITATIONS
SEE PROFILE

Thierry Siméon
Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS)
149 PUBLICATIONS **7,105** CITATIONS
SEE PROFILE

Enhancing sampling-based kinodynamic motion planning for quadrotors

Alexandre Boeuf, Juan Cortés, Rachid Alami and Thierry Siméon

Abstract— **The overall performance of sampling-based motion planning algorithms strongly depends on the use of suitable sampling and connection strategies, as well as on the accuracy of the distance metric considered to select neighbor states. Defining appropriate strategies and metrics is particularly hard when considering robot dynamics, which is required to treat constrained motion planning problems for quadrotors. This paper presents an accurate but computationally fast quasi-metric to determine the proximity of dynamic states of a quadrotor, and an incremental state-space sampling technique to avoid generating local trajectories that violate kinodynamic constraints. Results show that the integration of the proposed techniques in RRT-based and PRM-based algorithms can drastically decrease computing time, up to two orders of magnitude.**

I. INTRODUCTION

Sampling-based motion planning algorithms [1] explore the connectivity of the configuration space (or $\mathcal{C}$-space) by sampling configurations and attempting to connect them to build an underlying graph whose nodes and edges are feasible configurations and local paths, respectively. The sampling method and the steering method (or interpolation method) used to connect samples are therefore essential components of this type of algorithms. Most of these algorithms also rely on a notion of distance on the $\mathcal{C}$-space to define a connection policy, which is also determinant for the efficiency of the exploration. In the Rapidly-exploring Random Tree (RRT) algorithm [2] for instance, expansion bias towards the largest Voronoi regions is a direct consequence of the nearest node connection strategy. In Probabilistic Roadmap (PRM) based algorithms [3], significant efficiency improvements are obtained by prioritizing connections to nodes according to their distance to a sampled configuration. Since the topology of the space being explored is induced by the steering method, a suitable distance metric in the connection policy should be correlated to the actual length of the local paths, rather than using a simple Euclidean distance. Note that discussions on the importance of the distance metric have been recurrent since early work on sampling-based motion planning algorithms [4]. Indeed, this is an open problem that still motivates the development of new approaches [5].

The motion planning problem becomes even harder when dealing with under-actuated systems. The problem difficulty if often circumvented by using a decoupled approach. For instance, motion planning for quadrotors can be treated this way: In a first stage, a collision-free path is planned for the robot's bounding sphere. This path is then transformed into a trajectory that meets both continuity requirements and the differential constraints [6]. However, for very constrained problems, this approach is insufficient since there is no solution path for the bounding sphere. Several approaches have been proposed to generate local trajectories of quadrotors [7], [8], but few works tackle the global kinodynamic motion planning problem [9], which is mandatory in those cases.

All authors are with CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France and Univ de Toulouse, LAAS, F-31400 Toulouse, France (e-mails: aboeuf@laas.fr, jcortes@laas.fr, alami@laas.fr, nic@laas.fr)

This work has been partially supported by the European Community under Contract ICT 287617 "ARCAS".

Defining appropriate sampling and connection strategies is indeed more difficult when considering the robot dynamics. Treating the kinodynamic problem implies to explore the state space (or $\mathcal{X}$-space) rather than the $\mathcal{C}$-space, which means also considering derivatives of the degrees of freedom. Finding the ideal metric in this case may be as hard as solving an optimal control problem.

This paper builds on our recent work on quadrotor motion planning [10], which presented a steering method to connect kinodynamic states of this type of robots, together with first results on its integration inside a simple RRT algorithm. The principle of the steering method is recalled in Section II. Here, we propose a (quasi-)distance metric and a state-space sampling strategy aimed to significantly improve the performance of motion planning algorithms making use of this type of steering method. Sections III and IV present these two contributions together with some results that validate their relevance. Then, results showing the influence of both the metric and the sampling strategy on the overall performance of RRT-based and PRM-based planners are presented in Section V.

II. STEERING METHOD

This section briefly recalls some notions about the state space of a quadrotor and gives an overview of the steering method presented in [10][1]. The steering method uses fourth-order splines to generate smooth trajectories between two arbitrary states of a quadrotor, ensuring continuity up to the jerk (and therefore, up to the angular velocity), which is important to facilitate controllability along the planned trajectories. These trajectories minimize flying time (with respect to the proposed closed-form solution) while respecting kinodynamic constraints.

A. *State space of a quadrotor*

A quadrotor is a free-flying object in $\mathbb{R}^3$. It has three degrees of freedom in translation (position of the center of mass: $[x, y, z] \in \mathbb{R}^3$) and three in rotation (roll, pitch and yaw angles: $[\theta, \phi, \psi]$). Thus, its configuration space

[1]The code (C and MATLAB versions) is available upon request.

is $\mathcal{C} = SE(3)$. Considering a standard dynamic model, the system is differentially flat for the outputs $[x, y, z, \psi]$ (see [8]). This implies that any trajectory in the state of the flat outputs is flyable if their derivatives are correctly bounded. We define a kynodynamic state as a vector $\mathbf{x} = [x, y, z, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\psi}, \ddot{x}, \ddot{y}, \ddot{z}, \ddot{\psi}] \in \mathcal{X}$. With $\mathcal{X}$ the state space:

$$\mathcal{X} = \mathcal{P} \times \mathcal{V} \times \mathcal{A}$$

where $\mathcal{P}$, $\mathcal{V}$ and $\mathcal{A}$ are intervals of $\mathbb{R}^4$. Note that although there are no actual limits (if friction is neglected) on linear velocity of the center of mass of a quadrotor, for safety reasons, it might be interesting to take such limits into consideration for trajectory planning.

B. Steering method

The steering method we consider provides a solution (S, T) to the problem:

$$\begin{cases} [S(0)\ \dot{S}(0)\ \ddot{S}(0)] = \mathbf{x}_0 \in \mathcal{X} \\ [S(T)\ \dot{S}(T)\ \ddot{S}(T)] = \mathbf{x}_T \in \mathcal{X} \\ \forall t \in [0, T] \begin{cases} \dot{S}(t) \in \mathcal{V} \\ \ddot{S}(t) \in \mathcal{A} \\ \dddot{S}(t) \in \mathcal{J} \\ \ddddot{S}(t) \in \mathcal{S} \end{cases} \end{cases} \tag{1}$$

where S is the trajectory in the state of the flat outputs, T is the total time of the motion and $\mathcal{V}$, $\mathcal{A}$, $\mathcal{J}$ and $\mathcal{S}$ are zero centered intervals of $\mathbb{R}^4$. Obtaining the time-optimal trajectory is difficult and computationally expensive. This is not well suited for integration in sampling-based motion planners since they make extensive use of the steering method (usually thousands of calls to solve constrained planning problems). Therefore, we propose a method to compute a trajectory that approaches the optimal one with a simpler (imposed) shape enabling a rapid, analytical solution. The method works in two stages. First a solution is computed independently for each output x, y, z and ψ. Each solution is a fourth order spline $s(t)$ with a bang-null snap profile (i.e. a trapezoidal jerk profile) such that $[s(0)\ \dot{s}(0)\ \ddot{s}(0)] = [x_{0i}\ v_{0i}\ a_{0i}]$ and $[s(T_i)\ \dot{s}(T_i)\ \ddot{s}(T_i)] = [x_{Ti}\ v_{Ti}\ a_{Ti}]$ with $T_i \in \mathbb{R}^+$ (different for each output of index i). These splines are divided into seven main temporal phases (see Fig. 1). Four of them (noted A, C, E and H) correspond to acceleration variations. Phases B and G are constant acceleration phases. The goal is to maximize acceleration variations in order to reach full speed as fast as possible and maintain it as long as possible. Solutions are then synchronized to form the local trajectory whose total duration is $T = \max_{i=1..4} T_i$. Note that this method can be generalized to various types of systems. For a serial manipulator, for instance, each degree of freedom would be treated as a flat output.

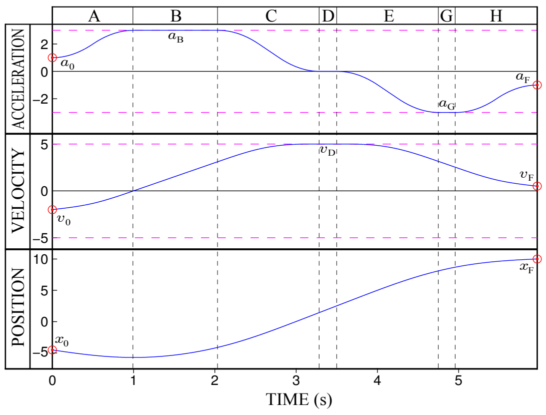

Fig. 1. Illustration of the seven main temporal phases of the spline for one output. Red circled dots are the values to be interpolated. Pink dashed lines are the boundary values for acceleration and velocity.

III. QUASI-METRIC IN THE STATE-SPACE

A. Motivations

The efficiency of the state-space exploration using randomized kinodynamic motion planning algorithms relies on a good distance metric. However, as discussed in [9], computing the actual distance between two states is as hard (and thus as costly) as solving the corresponding optimal control problem. Our steering method provides a deterministic suboptimal solution to such a control problem. Therefore, it defines a quasi-metric[2] $M^*_{SM} : (\mathbf{x}_0, \mathbf{x}_T) \mapsto T$ on the state space. Because of the dynamics of the system, a trajectory in the state space from $\mathbf{x}_0$ to $\mathbf{x}_T$ is indeed necessarily different from a trajectory from $\mathbf{x}_T$ to $\mathbf{x}_0$, and thus M^*_{SM} is not symmetric. Although this steering method is computationally fast, it is still too costly to be used for nearest neighbor search inside a sampling-based planner. This section presents a method to approximate the quasi-metric M^*_{SM} at a very low computational cost, and presents results that show its relevance.

B. Approximate quasi-metric

The complexity of the problem (1) defined in Section II is mainly due to its order (four) and to the inequality constraints on the derivatives. We propose to solve a simpler time optimal control problem for the third order (i.e. by considering the jerk as the control input), in one dimension and without constraints other than the bounds on the control input. The problem is then to find for each output of index i the couple (S_i, T_i) such that:

$$\begin{cases} \min T_i \in \mathbb{R}^+ \text{ s.t.} \\ [S_i(0)\ \dot{S}_i(0)\ \ddot{S}_i(0)] = [x_0\ v_0\ a_0] \in \mathbb{R}^3 \\ [S_i(T_i)\ \dot{S}_i(T_i)\ \ddot{S}_i(T_i)] = [x_{T_i}\ v_{T_i}\ a_{T_i}] \in \mathbb{R}^3 \\ \forall t \in [0, T_i],\ |\dddot{S}_i(t)| \le j_{max} \in \mathbb{R}^+ \end{cases} \tag{2}$$

Pontryagin maximum principle (see for example [11]) says that the optimal control is necessarily saturated, i.e.:

$$\forall t \in [0, T_i],\ \dddot{S}_i(t) \in \{-j_{max}, j_{max}\}$$

with at most two control commutations. Solving (2) implies to find T_i and these (at most) two commutation times, which requires to solve polynomial equations of maximum degree four. Further details can be found

[2]A quasi-metric has all the properties of a metric, symmetry excepted.

at http://homepages.laas.fr/aboeuf/iros15annex.pdf. The proposed quasi-metric is then defined as:

$$M_{SM} : (\mathbf{x}_0, \mathbf{x}_T) \mapsto \max_{i=1..4} T_i$$

C. Results

Here we present results of an experimental test to validate the proposed approximate quasi-metric. 10^4 pairs of kinodynamic states were randomly sampled in $\mathcal{X} = [-5,5]^3 \times [-5,5]^3 \times [-10,10]^3$, considering $\mathcal{J} = [-20,20]^3$ and $\mathcal{S} = [-50,50]^3$. Note that, without loss of generality and for simplification purposes, we consider here a constant yaw. For each pair $(\mathbf{x}_1, \mathbf{x}_2)$, we computed the value $M^*_{SM}(\mathbf{x}_1, \mathbf{x}_2)$ of the quasi-metric induced by our steering method, the value $M_{SM}(\mathbf{x}_1, \mathbf{x}_2)$ given by the proposed approximation, and the value $ED(\mathbf{x}_1, \mathbf{x}_2)$ of the euclidean distance in $\mathbb{R}^3$ considering only the position of the center of mass. We study the distribution of the relative error between M^*_{SM} and M_{SM}, i.e. the quantity:

$$RE_{M_{SM}}(\mathbf{x}_1, \mathbf{x}_2) = 1 - \frac{M_{SM}(\mathbf{x}_1, \mathbf{x}_2)}{M^*_{SM}(\mathbf{x}_1, \mathbf{x}_2)}$$

For comparison, we also provide the relative error RE_{ED} between M^*_{SM} and ED. Fig. 2 shows histograms of the distributions of these errors, Tab. I shows key statistical values of these distributions and Tab. II gives mean CPU times in milliseconds for a single core of an Intel Xeon W3520 processor at 2.67GHz.

The low standard deviation of the distribution of the relative error for the proposed quasi-metric is a measure of the quality of the approximation. These results also provide empirical evidence that M_{SM} and M^*_{SM} are equivalent since for all pairs $(\mathbf{x}_1, \mathbf{x}_2)$,

$$0.16396 \leq RE_{M_{SM}}(\mathbf{x}_1, \mathbf{x}_2) \leq 0.85540$$

which implies

$$\frac{1}{10}.M^*_{SM}(\mathbf{x}_1, \mathbf{x}_2) < M_{SM}(\mathbf{x}_1, \mathbf{x}_2) < 10.M^*_{SM}(\mathbf{x}_1, \mathbf{x}_2)$$

This means that M_{SM} and M^*_{SM} are inducing the same topology on $\mathcal{X}$ which means that the cost-to-go defined by our steering method is correctly evaluated by M_{SM}. This is clearly not the case for ED.

IV. SAMPLING STRATEGY

This section presents an incremental state-space sampling technique that increases the probability of generating connectible states. The definition of such states is first presented together with our motivations. Then the different steps of the method are presented. Finally some results are provided.

TABLE I
DISTRIBUTIONS OF THE RELATIVE ERRORS

Metric	M_{SM}	ED
Minimum	0.16396	−4.67050
Maximum	0.85540	0.94781
Mean	0.35918	−0.59440
Median	0.32806	−0.52272
Standard deviation	0.10308	0.69674

TABLE II
MEAN CPU TIMES IN MILLISECONDS

M^*_{SM}	M_{SM}	ED
1.23×10^{-1}	5.81×10^{-3}	1.10×10^{-4}

A. Connectible states and motivations

Trajectories generated by the steering method presented in Section II do not guarantee to respect bounds on the position of the robot. Such a constraint is typically violated when samples are close to the boundary of the workspace and the velocity is high, so that it is not possible to decelerate to avoid crossing this positional limit. In a similar way, bounds on velocity can also be violated. If acceleration is too high and velocity is close to the limit, produced trajectories will be invalid because velocity can not be reduced in time to meet the constraints. Note however that the imposed shape for the trajectories produced by our steering method guarantees that bounds on acceleration are respected.

We say that a kinodynamic state $\bar{\mathbf{x}} \in \mathcal{X}$ is *forward-connectible* (respectively *backward-connectible*) if and only if there is a state $\mathbf{x} \in \mathcal{X}$ such that the local path $(\bar{\mathbf{x}}, \mathbf{x})$ (respectively $(\mathbf{x}, \bar{\mathbf{x}})$) produced by the steering method lies entirely in $\mathcal{X}$ (i.e. respects bounds on position, velocity and acceleration). A state that is forward-connectible and backward-connectible is said to be connectible (non-connectible otherwise). This is illustrated on Fig. 3.

In case of uniform sampling of the state space, non-connectible states can be generated, and thus, local paths computed to connect those states have to be discarded *a posteriori* by the planner. This is rather inefficient since generating and testing a local path for validity is a costly operation. The goal of the sampling technique proposed below is to notably reduce the probability of generating non-connectible states, and hence to improve the performance of planning algorithms.

The sampling technique proceeds in a decoupled and incremental way. First, acceleration is uniformly sampled. The idea is then to compute a set of velocity values for which the state is known to be non-connectible. Velocity is then uniformly sampled outside this set. Finally, given this couple (velocity, acceleration) a set of position values for which the state is known to be non-connectible is computed, and the position is then uniformly sampled outside this set.

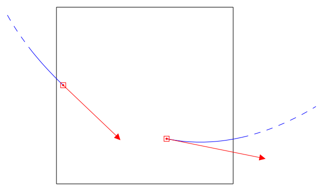
Fig. 3. Examples of non-connectible states in two dimensions. Red squares are positions and red arrows are velocity vectors. Blue curves are examples of trajectories. Bounds on position are represented in black. The state on the left is not backward-connectible. The state on the right is not forward-connectible.

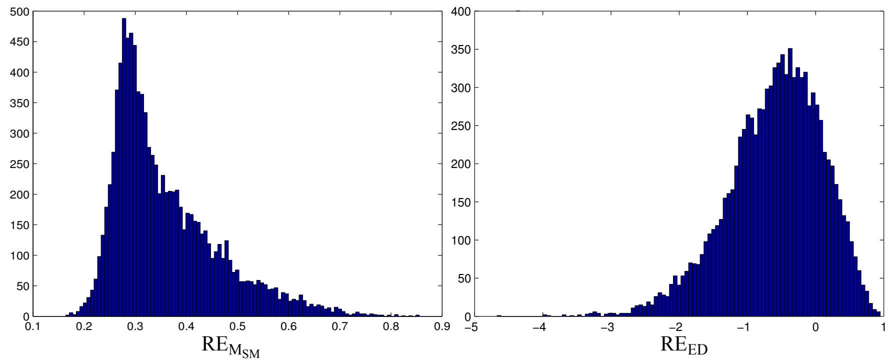

Fig. 2. Histograms of the distributions of the relative errors

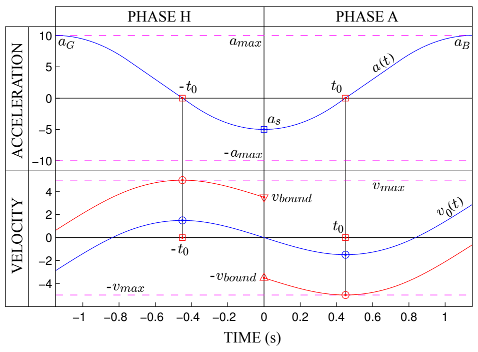

Fig. 4. Acceleration and velocity (blue curves) around $t = 0$. Saturated acceleration variation is applied in order to determine the limits on initial velocity (red triangles). Red curves are velocities for these limits.

B. Sampling velocity

This subsection explains how to compute the set of velocity values for which the state is known to be non-connectible, given a uniformly sampled acceleration value a_s. Explanations are given for one output. Fig. 4 illustrates this explanation. Let us denote v_{max} and a_{max} as the bounds on the absolute value of velocity and acceleration respectively. We study acceleration $a(t)$ and velocity $v_0(t)$ on a neighborhood around $t = 0$ for $a(0) = a_s$ and $v_0(0) = 0$. The idea is to apply a saturated acceleration variation and determine the extrema of $v_0(t)$ in this neighborhood. Using them, we can compute the limits on velocity v_s such that $v(t) = v_0(t) + v_s$ lies in $[-v_{max}, v_{max}]$. We use notations defined in section II. For $t > 0$, phase A of our steering method is applied. Phase H is applied for $t < 0$. The sampled value a_s locally imposes a direction of variation of $v_0(t)$ on phases A and H. We want to reverse this direction of variation in minimum time. This is equivalent to driving $a(t)$ to zero in minimum time. For that, we set $a_B = a_G = -\mathrm{sign}(a_s).a_{max}$. This corresponds to the highest acceleration variation achievable by our steering method. Note that, by construction, acceleration is symmetric during phases A and H (i.e $a(-t) = a(t)$) and $v_0(t)$ is anti-symmetric (i.e $v_0(-t) = -v_0(t)$). Since $a(t)$ is a second order spline strictly monotonic on phase A, it is straightforward to compute the unique $t_0 > 0$ such that $a(t_0) = 0$. The value $v_{bound} = v_{max} - |v_0(t_0)|$ is then the upper bound on the absolute value of v_s. This means that if $|v_s| > v_{bound}$ then $v(t) = v_0(t) + v_s$ will violate the constraints on velocity. A velocity value v_s is then uniformly sampled in $[-v_{bound},\ v_{bound}]$.

C. Sampling position

Given a couple (v_s, a_s) for one output, this subsection explains how to compute the set of position values for which the state is known to be non-connectible. Fig. 5 illustrates this explanation. The principle is similar to the one in the previous subsection. Velocity $v(t)$ and position $x_0(t)$ are studied around $t = 0$ for $a(0) = a_s$, $v(0) = v_s$ and $x_0(t) = 0$. We apply a saturated velocity variation and determine the extrema of $x_0(t)$ in this neighborhood. Using them, we can compute the limits on position x_s such that $x(t) = x_0(t)+x_s$ lies in $[-x_{max}, x_{max}]$ (bounds on position). For $t > 0$, phases A to C of our steering method are applied. Phases E to H are applied for $t < 0$. We want to reverse the direction of variation of the position imposed by v_s as fast as possible. This is equivalent to driving $v(t)$ to zero in minimum time. For that, we set $v_D = -\mathrm{sign}(v_s).v_{max}$ for both phases A to C and E to H. This corresponds to the highest velocity variation achievable by our steering method. The only difference here is that neither $v(t)$ nor $x_0(t)$ have symmetry proprieties. We compute $t^+ > 0$ such that $v(t^+) = 0$ and $t^- < 0$ such that $v(t^-) = 0$. If $v_s \geq 0$ then $x^+ = x_{max} - x_0(t^+)$ and $x^- = -x_{max} - x_0(t^-)$ else $x^+ = x_{max} - x_0(t^-)$ and $x^- = -x_{max} - x_0(t^+)$. A position value x_s is then uniformly sampled in $[x^-,\ x^+]$.

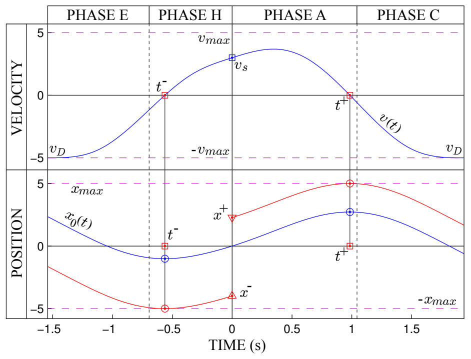

Fig. 5. Velocity and position (blue curves) around $t = 0$. Saturated velocity variation is applied in order to determine the limits on initial positions (red triangles). Red curves are positions for these limits.

D. Results

We provide here some results concerning the sampling strategy. The conducted experiment consisted in testing the validity of local paths computed between uniformly sampled pairs of states in $\mathcal{X} = [-5,5]^3 \times [-5,5]^3 \times [-10,10]^3$, with $\mathcal{J} = [-20,20]^3$ and $\mathcal{S} = [-50,50]^3$. Yaw was kept constant. We measured the percentage of valid paths (i.e. which lies entirely in $\mathcal{X}$) over 10^4 calls. We then repeated this operation using our sampling technique. With uniform sampling only 11.53% of the produced paths were valid. With our sampling technique 95.58% of the paths were valid. Moreover, we can test for a given state if each output is respecting the constraints defined by our sampling technique, i.e. if velocity lies in $[-v_{bound},\ v_{bound}]$ and if position lies in $[x^-,\ x^+]$. When this is not the case we consider that the state is not connectible. Our sampling technique is obviously generating connectible states every time whereas with uniform sampling about 90% of the generated states are not connectible (with respect to this criteria).

V. INFLUENCE ON PLANNING ALGORITHMS

This section describes variants of RRT and PRM algorithms, and provides some results on the influence of both the proposed quasi-metric and the sampling strategy on them. These algorithms are well known, but had to be adapted to the non-symmetry of the steering method and the associated quasi-metric. Because of this non-symmetry, the underlying graph is directed.

A. Directed bi-RRT

In the well known undirected version of the bi-RRT algorithm [2], two trees $\mathcal{T}_S$ and $\mathcal{T}_G$ are constructed in parallel. $\mathcal{T}_S$ grows from the start configuration and $\mathcal{T}_G$ from the goal configuration. Each iteration for one of the trees consists of sampling a configuration q_{rand}, finding its nearest neighbor q_{near} in the tree (according to a defined metric), and extending it toward q_{rand} (using a steering method) to create a new configuration q_{new}. Each time an expansion is successful for one of the trees, a direct connection is attempted between q_{new} and its nearest neighbor in the other tree. The algorithm ends if this local path is valid (i.e. when the trees are connected). In our directed version, both the steering method and the quasi-metric M_{SM} are non-symmetric, and thus have to be called taking care of the order of the two states. The nearest neighbors $N_S(\bar{\mathbf{x}})$ and $N_G(\bar{\mathbf{x}})$ of a state $\bar{\mathbf{x}}$ in $\mathcal{T}_S$ and $\mathcal{T}_G$ respectively are defined as such:

$$\begin{cases} N_S(\bar{\mathbf{x}}) = \underset{\mathbf{x}\in\mathcal{T}_S}{\arg\min}\, M_{SM}(\mathbf{x},\bar{\mathbf{x}}) \\ N_G(\bar{\mathbf{x}}) = \underset{\mathbf{x}\in\mathcal{T}_G}{\arg\min}\, M_{SM}(\bar{\mathbf{x}},\mathbf{x}) \end{cases}$$

For an expansion of $\mathcal{T}_S$, we test the local path $\big(N_S(\mathbf{x}_{rand}),\mathbf{x}_{new}\big)$ for validity. In case of success, the algorithm ends if the local path $\big(\mathbf{x}_{new},N_G(\mathbf{x}_{new})\big)$ is valid. For an expansion of $\mathcal{T}_G$, the local path $\big(\mathbf{x}_{new},N_G(\mathbf{x}_{rand})\big)$ is tested for validity, and the algorithm ends in case of validity of the local path $\big(N_S(\mathbf{x}_{new}),\mathbf{x}_{new}\big)$.

B. Directed PRM

At each iteration of the undirected version of the PRM algorithm [3], a collision free configuration q is sampled and added to the graph $\mathcal{G}$. For every connected component G_i of $\mathcal{G}$, connections are attempted between q and each node of G_i in increasing order of distance from q until one is successful. A threshold on this distance can be considered with the aim to reduce computational cost. In our directed version, we consider the strongly connected components G_i of $\mathcal{G}$. Moreover, we maintain during the execution the adjacency matrix $A_{\mathcal{G}}$ of the transitive closure of the graph of the strongly connected components of G. This square matrix, whose dimension is the number of strongly connected components, is defined by $A_{\mathcal{G}}[i][j] = 1$ if a path in $\mathcal{G}$ exists from every node of G_i to every node of G_j and $A_{\mathcal{G}}[i][j] = 0$ otherwise. If $A_{\mathcal{G}}[i][j] = 1$ we say that G_i is connected to G_j. Note that $A_{\mathcal{G}}[i][j] = A_{\mathcal{G}}[j][i] = 1$ if and only if $i = j$. At each iteration, a valid state $\mathbf{x}$ is sampled and added to $\mathcal{G}$ (which has n strongly connected components). Its strongly connected component $G_{n+1} = \{\mathbf{x}\}$ is added to the matrix $A_{\mathcal{G}}$. For every connected component G_i of $\mathcal{G}$ ($i = 1..n$), if G_i is not connected to G_{n+1}, connections from every node $\mathbf{x}_j$ of G_i to $\mathbf{x}$ are attempted in increasing order of $M_{SM}(\mathbf{x}_j,\mathbf{x})$ until one is valid. As for the undirected version, a threshold on the value of M_{SM} can be considered here. $A_{\mathcal{G}}$ is updated if neecessary. Then, if G_{n+1} is not connected to G_i, connections from $\mathbf{x}$ to every node $\mathbf{x}_j$ of G_i are attempted in increasing order of $M_{SM}(\mathbf{x},\mathbf{x}_j)$ until one is valid. If used in single query mode, the algorithm ends when the strongly connected component of the initial state is connected to the strongly connected component of the goal state.

C. Results

Results presented below show the influence of the quasi-metric and the sampling technique on the two previously presented motion planners. Experiments have been conducted on two different environments shown in Fig. 6 and for the same quadrotor whose diameter is equal to 0.54 meters. We consider $\mathcal{V} = [-5,5]^3$, $\mathcal{A} = [-10,10]^3$, $\mathcal{J} = [-20,20]^3$, $\mathcal{S} = [-50,50]^3$ (using SI base units). Yaw is kept constant. The first environment, referred to as *boxes*, is a cube with side length of 10 meters filled with box shaped-obstacles of different sizes. The second environment, referred to as *slots*, is also a cube with side length of 10 meters but divided in

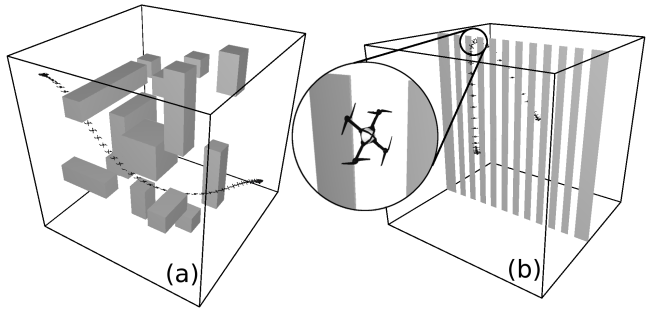

Fig. 6. Testing environments (a) *boxes* (b) *slots*

TABLE III

B: BOXES, S: SLOTS, P: PRM, R: RRT, M: PROPOSED METRIC, E: EUCLIDEAN METRIC, I: INCREMENTAL SAMPLING, U: UNIFORM SAMPLING

Experiment	B.P.M.I.	B.P.E.I.	B.P.M.U.	B.P.E.U.	S.P.M.I.	S.P.E.I.	S.P.M.U.	S.P.E.U.
CPU time (s)	0.05648	0.07884	3.284	4.409	1.505	1.220	578.5	444.2
Flying time (s)	8.180	8.772	8.000	8.126	9.074	8.979	8.615	8.387
Number of nodes	12.11	13.77	78.64	88.33	71.93	61.59	767.9	725.8
% of not connectible nodes	0	0	82.53	84.38	0	0	89.11	89.19

Experiment	B.R.M.I.	B.R.E.I.	B.R.M.U.	B.R.E.U.	S.R.M.I.	S.R.E.I.	S.R.M.U.	S.R.E.U.
CPU time (s)	0.02780	0.04088	0.04144	0.05612	2.165	2.466	558.8	512.9
Flying time (s)	9.674	10.84	9.365	10.09	25.42	34.72	33.96	55.98
Number of nodes	8.79	8.84	9.18	10.77	334.5	565.6	4429.4	8813.0
Number of iterations	26.45	45.04	45.58	65.02	982.9	2502.9	30253.7	196233.3
% of not connectible nodes	50.34	54.94	51.51	59.85	25.60	34.86	79.41	82.48

two halves by a series of aligned obstacles separated by 0.40 meters (hence smaller than the robot diameter). This problem is particularly challenging since going across these obstacles requires to find a path in a very narrow passage in the state-space. Every combination of environment, algorithm, metric and sampling strategy has been tested. Results are provided in Tab. III for CPU and flying times in seconds, number of nodes (and iterations for the RRT) and percentage of not connectible nodes (with respect to the criteria defined in section IV.D.). Each experiment is designated by an acronym whose meaning is explained in the caption. Results are averaged over 100 runs and are for an implementation in C, integrated in our motion planning software Move3D [12], and run on a single core of an Intel Xeon W3520 processor at 2.67GHz.

Results show a significant improvement of the performance of both algorithms thanks to the integrations of the proposed techniques. However, one can clearly see that the metric and the sampling technique have a more notable effect on one or the other planner. Results for the PRM algorithm shows that the sampling method has a great influence on its performance. Its integration indeed improves CPU time by two orders of magnitude for both environments. On the other hand, one can see that for the *slots* environment CPU times are slighlty worse with the use of the metric. This can be explained by the difference of computing time between our quasi-metric and the euclidean distance. This is also observed in one case for the RRT algorithm (SRMU vs. SREU). For the RRT algorithm, results show that the influence of the metric is more important. This was to be expected since RRT-based algorithms are known to be very sensitive to the metric. One can see that the number of iterations is significantly reduced, meaning that the search is better guided. The improvement produced by the sampling technique is also very significant for the *slots* environment but less noticeable for the *boxes* environment. This can be explained by the fact that, in RRT-based algorithms, the sampled states are not tested for connections but used to define a direction for extension. A new state is then generated according to that direction. One can see that, for the *boxes* environment, about half of these states are not connectible regardless of the sampling method. Finally note that flying times are given for the raw, non-smoothed trajectories, which explains the rather large difference of path quality between PRM and RRT results.

VI. CONCLUSION

We have presented two techniques to enhance the performance of kinodynamic motion planning algorithms applied to quadrotors: 1) a quasi-metric in the state space that accurately estimates the actual quasi-distance induced by a steering method, but which is two orders of magnitude less computationally expensive; 2) An incremental state-space sampling technique that notably increases the probability of generating connectible states. Results have show that the integration of these techniques can drastically change the performances of PRM-based and RRT-based planners, particularly when solving very constrained problems.

We are currently working on the comparison of the trajectories generated by the proposed steering method and those obtained by a numerical optimal control method. First results tend to show the good quality of our approximate solution, which is orders of magnitude faster to compute. We are also working on the integration of the planning methods in an indoor testbed for experimental validation.

REFERENCES

[1] S. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[2] S. LaValle and J. Kuffner, "Rapidly-exploring random trees: progress and prospects," in *Algorithmic and Computational Robotics: New Directions*. A. K. Peters, Wellesley, MA, 2001.

[3] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation, 12(4)*, pp. 566–580, 1996.

[4] N. Amato, O. Bayazit, L. Dale, C. Jones, and D. Vallejo, "Choosing good distance metrics and local planners for probabilistic roadmap methods," *IEEE Transactions on Robotics and Automation, 16(4)*, pp. 442–447, 2000.

[5] M. Bharatheesha, W. Caarls, W. J. Wolfslag, and M. Wisse, "Distance metric approximation for state-space RRTs using supervised learning," in *Proc. IEEE/RSJ IROS*, 2014.

[6] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Proc. ISRR*, 2013.

[7] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification," in *Proc. IEEE/RSJ IROS*, 2013.

[8] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE ICRA*, 2011.

[9] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research, 20(5)*, pp. 378–400, 2001.

[10] A. Boeuf, J. Cortés, R. Alami, and T. Siméon, "Planning agile motions for quadrotors in constrained environments," in *Proc. IEEE/RSJ IROS*, 2014.

[11] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, 2005, vol. I.

[12] T. Siméon, J.-P. Laumond, and F. Lamiraux, "Move3D: a generic platform for path planning," in *Proc. IEEE ISATP*, 2001.