2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

Sensor-based Exploration for General Robotic Systems

Luigi Freda Giuseppe Oriolo Francesco Vecchioli
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Ariosto 25, 00185 Roma, Italy
{freda,oriolo}@dis.uniroma1.it, francesco_vecchioli@fastwebnet.it

Abstract— **We present a method for sensor-based exploration of unknown environments by a robotic system equipped with rangefinders. The method is based on the incremental generation of a configuration-space data structure called Sensor-based Exploration Tree (SET). The expansion of the SET is driven by information at the world level, where the perception process takes place. In particular, the frontiers of the explored region are used to guide the search for informative view configurations. Various exploration strategies may be obtained by instantiating the general SET method with different sampling techniques. Two of these are compared by simulations in 2D and 3D worlds.**

I. INTRODUCTION

In service applications, robotic systems are often required to carry out given tasks in environments that are partially or completely unknown. Sensing, planning and motion execution must then be appropriately interlaced in order to discover and navigate the portions of the environment that are relevant to the assigned task. For example, *sensor-based motion planning* addresses the problem of finding a collision-free path from a start configuration to a goal configuration in an unknown environment [1], [2].

A related but different scenario is *sensor-based exploration*, in which the problem is to 'cover' the environment as much as possible with sensory perceptions. Often, this is aimed at building a map of the environment, which can then be used to plan and execute further actions; however, this may not be required, e.g., if the objective is simply to locate a lost object.

A consistent amount of literature deals with sensor-based exploration using single-body mobile robots. Typically, it is assumed that the robot is disk-shaped and equipped with an omnidirectional laser rangefinder. For this problem, there exist many exploration algorithms which fall into the class of *frontier-based strategies* [3]–[7]. These are based on the idea that the robot should approach the boundary between explored and unexplored areas of the environments in order to maximize the expected utility of robot motions.

The problem of exploring an unknown world with a multi-body robotic system, such as a fixed or mobile manipulator, is more challenging. This is essentially due to the fact that the sensing space (the world) and the planning space (the configuration space) are very different in nature. In particular, the former is a Euclidean space of dimension 2 or 3, while the latter is a manifold in the presence of angular coordinates and has dimension equal to the number of dof's of the robot, typically 6 or more. While frontiers at the world level clearly retain their informative value, using this information to plan actions in configuration space is not straightforward. In the literature, few works exist that address this problem mainly for fixed-base manipulators, e.g., see [8]–[11].

In this work, we present the SET (Sensor-based Exploration Tree) method, a frontier-based exploration strategy for general robotic systems that can be seen as an extension of the method for mobile robots described in [7], [12]. The basic idea is to guide the robot so as to bring the sensory system to perform a depth-first exploration of the world, progressively sensing regions that are contiguous from the viewpoint of sensor location. The information gathered about the free space is mapped to a configuration space roadmap which is built via a sampling procedure. The latter is used to select a new view configuration, which is added to the SET. In the exploration process, the robot alternates forwarding/backtracking motions on the SET, which essentially acts as an Ariadne's thread.

Two main exploration strategies can be obtained instantiating the proposed general method with different sampling techniques. In the first, the roadmap is expanded using a global sampling approach. In the second, the roadmap is built in the form of a forest of connected trees, each rooted at a distinct view configuration and grown through a 'local' sampling procedure. The two proposed strategies are compared by simulations using various robots in different scenes.

The paper is organized as follows. The problem setting is given in Sect. II. A general exploration strategy is outlined in Sect. III, and the useful concept of free boundary is introduced in Sect. IV. Based on this, the SET method is presented in Sect. V. Simulation results in different worlds are reported and discussed in Sect. VI. Some extensions of the present work are mentioned in the concluding section.

II. GENERAL PROBLEM SETTING

The robot 'wakes up' in a unknown world populated by obstacles. It has to explore an build a model of the world. Exploration is performed by 'covering' the world as much as possible with sensory perceptions.

We start with a problem formulation suitable for general

This work has been funded by the European Commission's Sixth Framework Programme as part of the project PHRIENDS under grant no. 045359.

robotic systems. This setting will be particularized to a specific case when describing the proposed exploration strategy.

A. Robot and World Models

The robot is called $\mathcal{R}$. It consists of a kinematic chain of r rigid bodies ($r \geq 1$) interconnected by elementary joints. This description includes: fixed-base manipulators, single-body mobile robots and multiple-body mobile robots, such as vehicles with trailers, snake-like robots, humanoids and mobile manipulators.

The *world* $\mathcal{W}$ is a compact subset of $\mathbb{R}^N$, with $N = 2$ or 3. It represents the physical space in which the robot moves and perceives. $\mathcal{W}$ contains static obstacles $\mathcal{O}_j$, $j = 1, 2, ..., p$, each one a compact connected subset of $\mathcal{W}$. The boundary $\partial\mathcal{W}$ is assumed to act as a 'fence' and is therefore considered as an obstacle. The *obstacle region* $\mathcal{O}$ is the union of $\partial\mathcal{W}$ and all the obstacles $\mathcal{O}_j$, $j = 1, 2, ..., p$. The *free world* is $\mathcal{W}_{\text{free}} = \mathcal{W} \setminus \mathcal{O}$.

The robot *configuration space* is denoted by $\mathcal{C}$ and a robot configuration by q. Let $\mathcal{R}(q)$ be the compact region of $\mathcal{W}$ occupied by the robot at q. Define the *C-obstacle region* $\mathcal{CO}$ as the set of configurations q such that $\mathcal{R}(q) \cap \mathcal{O} \neq \emptyset$. The *free configuration space* is $\mathcal{C}_{\text{free}} = \mathcal{C} \setminus \mathcal{CO}$.

B. Sensor Model

The robot is equipped with a system of m exteroceptive sensors, whose operation is formalized as follows.

Assuming[1] that the robot is at q, denote by $\mathcal{F}_i(q) \subset \mathbb{R}^N$ the compact region occupied by the i-th sensor *field of view*, which is assumed to be star-shaped with respect to the sensor *center* $s_i(q) \in \mathcal{W}$. For instance, in $\mathbb{R}^2$, $\mathcal{F}_i(q)$ can be a circular sector with apex $s_i(q)$, opening angle α_i and radius R_i, where the latter is the perception range (see Fig. 1, left). The (total) *sensor field* at q is $\mathcal{F}(q)$, defined as the union of all the $\mathcal{F}_i(q)$ for $i = 1, 2, ..., m$.

Given a robot configuration q, a point $p \in \mathcal{W}$ is said to be *visible from the i-th sensor* if $p \in \mathcal{F}_i(q)$ and the open line segment joining p and $s_i(q)$ does not intersect $\partial\mathcal{O} \cup \partial\mathcal{R}(q)$. At each configuration q, the robot sensory system returns (see Fig. 1 right):

- the *visible free region* (or *view*) $\mathcal{V}(q)$, which collects all the points of $\mathcal{W}_{\text{free}}$ that are visible from at least one sensor,
- the *visible obstacle boundary* $\mathcal{B}(q) = \partial\mathcal{O} \cup \partial\mathcal{V}(q)$, which collects all the points of $\partial\mathcal{O}$ that are visible from at least one sensor.

The above sensor is an idealization of a 'continuous' rangefinder. In practice, such a sensor may be realized by a rotating laser rangefinder, which returns the distance to the nearest obstacle point along all the directions (*rays*) contained in its field of view (with a certain resolution). Another sensory system which satisfies the above description is a stereoscopic camera.

[1] In what follows, it is assumed that the placement of the sensor center is uniquely determined by the robot configuration q. If the sensor is not rigidly attached to the robot (e.g., if it can autonomously rotate around a certain 2D or 3D axis, or is mounted on a pan-tilt platform), it is then necessary to include the corresponding dof's in vector q.

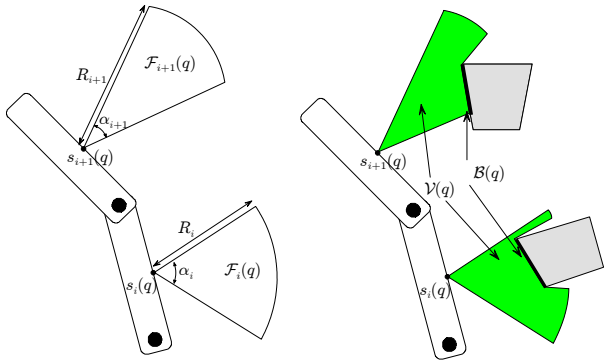

Fig. 1. *Left*: sensor centers $s_i(q)$ and $s_{i+1}(q)$, and the associated fields of view $\mathcal{F}_i(q)$ and $\mathcal{F}_{i+1}(q)$ when the robot is at configuration q. *Right*: The view $\mathcal{V}(q)$ and the visible obstacle boundary $\mathcal{B}(q)$.

C. Exploration task

The robot explores the world through a sequence of view-plan-move actions. Each configuration where a view is acquired is called a *view configuration*. Let q^0 be the initial robot configuration and $q^1, q^2, ..., q^k$ the sequence of view configurations assumed by the robot up to the k-th exploration step. When the exploration starts, all the initial robot endogenous knowledge can be expressed as

$$\mathcal{E}^0 = \mathcal{R}(q^0) \cup \mathcal{V}(q^0), \tag{1}$$

where $\mathcal{R}(q^0)$ represents the free volume that the robot body occupies (computed on the basis of proprioceptive sensors) and $\mathcal{V}(q^0)$ is the view at q^0 (provided by exteroceptive sensors). At step $k \geq 1$, the *explored region* is

$$\mathcal{E}^k = \mathcal{V}(q^k) \cup \mathcal{E}^{k-1}.$$

Since an obvious requirement is $\mathcal{R}(q^k) \subset \mathcal{E}^{k-1}$ for any k, we have

$$\mathcal{E}^k = \mathcal{R}(q^0) \cup \left(\bigcup_{i=0}^{k} \mathcal{V}(q^i) \right).$$

A point $p \in \mathcal{W}_{\text{free}}$ is defined *explored at step k* if it is contained in $\mathcal{E}^k$ and *unexplored* otherwise. At each step k, $\mathcal{E}^k \subseteq \mathcal{W}_{\text{free}}$ is the current estimate of the free world.

A configuration q is *safe at step k* if $\mathcal{R}(q) \subset \mathcal{E}^k$. The *safe region* $\mathcal{S}^k$ collects all the configurations that are safe at step k. Note that $\mathcal{S}^k \subseteq \mathcal{C}_{\text{free}}$ represents a configuration-space image of $\mathcal{E}^k$ and is the current estimate of $\mathcal{C}_{\text{free}}$. A path in $\mathcal{C}$ is *safe at step k* if it is completely contained in $\mathcal{S}^k$. The goal of the exploration is to expand $\mathcal{E}^k$ as much as possible as k increases.

III. EXPLORATION STRATEGIES

Assume the robot can associate an information gain $I(q, k)$ to any (safe) q at step k. This is an estimate of the world information which can be discovered by acquiring a view from q at the current step.

Consider the k-th exploration step, which starts with the robot at q^k. Let $\mathcal{Q}^k \subset \mathcal{S}^k$ be the *informative safe*

```
SET METHOD
1   update SET and world model
2   extract local free boundary LFB(q^k, k)
3   if local free boundary is not empty
4     (q^{k+1}, U(q^{k+1})) ← search configuration with maximum utility inside current admissible set D(q^k, k)   %search%
5   else
6     U(q^{k+1}) ← 0
7   if U(q^{k+1}) ≥ U_min                                                                                          %minimum utility check%
8     plan a safe path connecting q^k to q^{k+1}
9     move to q^{k+1} and acquire sensor view                                                                      %forwarding%
10  else
11    move to parent configuration                                                                                 %backtracking%
```

Fig. 2. A pseudocode description of the k-th iteration of the SET method in which the robot starts at q^k.

region, i.e., the set of configurations which have non-zero information gain and can be reached from q^k through a path that is safe at step k. In a general exploration strategy, the next view configuration q^{k+1} is chosen in $\mathcal{Q}^k \cap \mathcal{D}(q^k, k)$ according to some selection criterion (e.g., information gain maximization). The set $\mathcal{D}(q^k, k) \subseteq \mathcal{C}$ denotes an admissible set around q^k at step k: its shape and size determines the 'locality' of the search. For example, if $\mathcal{D}(q^k, k) = \mathcal{C}$, a global search is performed, whereas if $\mathcal{D}(q^k, k)$ is a small neighborhood of q^k the search is local in scope.

When $\mathcal{Q}^k \cap \mathcal{D}(q^k, k)$ is not empty, i.e., the admissible set at step k contains informative configurations, a safe motion is performed towards the new selected view configuration (*forwarding*). Otherwise, the robot returns back towards a previously visited view configuration (*backtracking*).

The exploration can be considered *completed* at step k if $\mathcal{Q}^k = \emptyset$, i.e., there is no configuration $q \in \mathcal{S}^k$ such that (1) a new view at q can extend the current explored region (2) the robot can reach q from q^k through a path which is safe at step k.

In order to completely characterize an exploration strategy, one needs to define *(i)* the set $\mathcal{D}(q^k, k)$ *(ii)* the information gain *(iii)* the selection strategy. In practice, due the complexity of the map from $\mathcal{E}^k$ to $\mathcal{S}^k$, it is also crucial to define an efficient procedure to compute $\mathcal{Q}^k \cap \mathcal{D}(q^k, k)$.

IV. THE FREE BOUNDARY

A useful tool, which provides information about $\mathcal{Q}^k$ without requiring its explicit computation, is the *free boundary* (also called *frontier*[2] in the literature [3]–[7]) of the explored region. At step k, the boundary of the explored region $\partial\mathcal{E}^k$ is the union of two *disjoint* sets:

- the *obstacle boundary* $\partial\mathcal{E}^k_{\text{obs}}$, i.e., the part of $\partial\mathcal{E}$ which is made by detected obstacle surfaces;
- the *free boundary* $\partial\mathcal{E}^k_{\text{free}}$, i.e., the complement of $\partial\mathcal{E}^k_{\text{obs}}$, which leads to potentially explorable areas.

The obstacle boundary $\partial\mathcal{E}^k_{\text{obs}}$ can be computed as the union of all the visible obstacle boundaries $\mathcal{B}(q^i)$, $i = 1, 2, ..., k$, collected by the robot up to the k-th step. The free boundary $\partial\mathcal{E}^k_{\text{free}}$ is then computed as $\partial\mathcal{E}^k \setminus \partial\mathcal{E}^k_{\text{obs}}$.

The free boundary has some useful properties. Given a safe configuration q, one has the following implications

$$\mathcal{F}(q) \cap \partial\mathcal{E}^k_{\text{free}} = \emptyset \;\Rightarrow\; I(q, k) = 0 \tag{2}$$

i.e., if no free boundary is present in the total field at q, then nothing can be discovered by acquiring a view from q. Moreover one has

$$\partial\mathcal{E}^k_{\text{free}} = \emptyset \;\Rightarrow\; \mathcal{Q}^k = \emptyset. \tag{3}$$

In fact, if the free boundary is empty, no more unexplored points remain in $\mathcal{W}_{\text{free}}$ and the exploration is complete. The converse of (2) and (3) is not true in general.

In the light of the above implications, and thanks to the low computational cost of the boundary, the central idea of the SET method is to guide the robot towards those configurations $q \in \mathcal{D}(q^k, k)$ at which $\mathcal{F}(q) \cap \partial\mathcal{E}^k_{\text{free}} \neq \emptyset$.

V. THE SET METHOD

In the SET Method, the robot incrementally builds the *Sensor-based Exploration Tree* (SET) data structure. Each node of the SET represents a view configuration, while an arc between two nodes represent a safe path joining the two view configurations.

An exploration cycle of the SET algorithm is shown in Fig. 2. We first give a quick commentary of these steps, and then discuss their structure in some detail.

The robot starts at q^k. First, the SET data structure and the world model are updated (line 1). In particular, if a new view configuration has been reached at the previous iteration: a new corresponding node is inserted in the SET, a new arc is created between the previous and the new node, the new acquired view is merged in the world model $(\mathcal{E}^k, \partial\mathcal{E}^k_{\text{obs}})$.

Next, the *local free boundary* $\text{LFB}(q^k, k)$ is extracted (line 2). Loosely speaking, it contains any portion of free boundary which is visible by the sensor from some $q \in \mathcal{D}(q^k, k)$ (see Sect. V-B for further details). It is defined so that it can be easily computed and the following implication holds:

$$\text{LFB}(q^k, k) = \emptyset \;\Rightarrow \mathcal{D}(q^k, k) \cap \mathcal{Q}^k = \emptyset \tag{4}$$

[2]In this paper, we prefer to avoid the word 'frontier' which can be easily confused with 'boundary'.

This means that $\mathrm{LFB}(q^k,k) \neq \emptyset$ is a necessary condition to be verified before searching for a view configuration in $\mathcal{D}(q^k,k) \cap \mathcal{Q}^k$ (line 4); if $\mathrm{LFB}(q^k,k) = \emptyset$ a backtracking must be forced. As detailed in Sect. V-C, the adopted selection strategy consists in the maximization of a defined utility function U in $\mathcal{D}(q^k,k)$. Two search strategies, which rely on sampling-based techniques, are proposed to this end (see Sect. V-D for further details). According to (4), when the local free boundary is empty, no search is performed and $U(q^{k+1})$ is set to zero forcing a backtracking (line 6).

At this point, the utility $U(q^{k+1})$ of the candidate view configuration q^{k+1} is compared with $U_{\min}$, a fixed minimum threshold (line 7). This check is twofold: 1) forces a backtracking step when the local free boundary is empty (i.e., when $U(q^{k+1})$ is set to zero) (2) filters almost useless robot actions. Hence, if $U(q^{k+1}) > U_{\min}$: q^{k+1} becomes the next view configuration, the path-planner is invoked to compute a path from q^k to q^{k+1} (line 8, see Sect. V-E for further details), q^{k+1} is reached and a new view is acquired (line 9). Otherwise, a backtracking step is performed and the robot moves to the parent configuration (line 11).

When the robot is unable to further 'push-forward' the visited local free boundaries, it is forced to backtrack to the SET root (the starting configuration) thus realizing an automatic *homing* mechanism.

A. Information Gain Definition

Let $\mathcal{V}(q,k)$ be the *simulated view* which would be acquired by the robot at q if the obstacle boundary were exactly $\partial\mathcal{E}^k_{\mathrm{obs}}$. Since our objective is to extend $\mathcal{E}^k$ as much as possible, we define the information gain $I(q,k)$ as the measure of the set of all the unexplored points lying in $\mathcal{V}(q,k)$. $I(q,k)$ can be computed by the robot through a ray-casting procedure.

Clearly, depending on the world representation and the robot task, other definitions of information gain are possible [13]. For instance, in the presence of sensor noise a probabilistic world representation can be adopted and entropy-based definitions of the free boundary and the information gain can be easily introduced.

B. Admissible Set and Local Free Boundary Definition

Since implication (4) is required to hold, the definitions of the admissible set and the local free boundary must be strictly related. In this subsection, we first define $\mathcal{D}(q^k,k)$, then $\mathrm{LFB}(q^k,k)$ is accordingly designed. For simplicity, we refer to the case of a robot with a single rangefinder ($m = 1$). Without loss of generality, we assume the field $\mathcal{F}(q)$ at q is a spherical cone with apex $s(q)$, radius R (perception range) and opening angle α.

1) Admissible Set: In the SET method, $\mathcal{D}(q^k,k)$ collects all the configurations q such that *(i)* the sensor center $s(q)$ is within a maximum distance ρ from $s(q^k)$ *(ii)* $s(q)$ and $s(q^k)$ are *mutually visible* at step k, i.e., the open line segment $(s(q), s(q^k))$ does not intersect $\partial\mathcal{E}^k_{\mathrm{obs}}$.

The first requirement, along with the imposed forwarding/backtracking algorithm structure, suggests a 'world-based depth-first' strategy. That is, the sensor center is guided so as to perform a depth-first traversal of the world, incrementally exploring contiguous regions. The second condition is a usual requirement in exploration of unknown environments. It prevents world exploration from continuously jumping between contiguous separated rooms (e.g., when $s(q^k)$ and $s(q^{k+1})$ are close but separated by a wall). Moreover, a collection of mutually visible sensing locations in the world can be seen as a visibility graph of recognizable known locations with interesting spatial and topological properties.

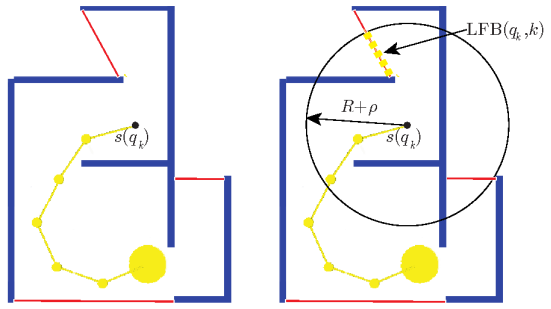

Fig. 3. A reconstructed world model at step k. *Left*: free boundary $\partial\mathcal{E}^k_{\mathrm{free}}$ (orange-thin) and obstacle boundary $\partial\mathcal{E}^k_{\mathrm{obs}}$ (blue-thick). The explored region $\mathcal{E}^k$ is enclosed by $\partial\mathcal{E}^k = \partial\mathcal{E}^k_{\mathrm{free}} \cup \partial\mathcal{E}^k_{\mathrm{obs}}$ *Right*: the local free boundary $\mathrm{LFB}(q^k,k)$ is highlighted by dashes.

2) Local Free Boundary: $\mathrm{LFB}(q^k,k)$ collects all the points of the free boundary $\partial\mathcal{E}^k_{\mathrm{free}}$ which *(i)* are contained in a ball $B(s(q^k),\rho+R)$ with center $s(q^k)$ and radius $\rho+R$ *(ii)* can be connected to $s(q^k)$ through a world path completely contained[3] in $\mathcal{E}^k \cap B(s(q^k),\rho+R)$ (see Fig. 3). Note that the parameter ρ is inherited from the admissible set definition.

It is easy to show that implication (4) holds for the definitions of $\mathrm{LFB}(q^k,k)$ and $\mathcal{D}(q^k,k)$ given above.

C. Selection Strategy

The most common approach to evaluate the contribution of a robot action towards the fulfillment of an assigned robot task is the association of a utility function. A maximization is then used to select the next action, possibly over a finite number of choices. Different requirements in the task can be translated in partial utility/cost terms which are combined in the total utility function U. An optimal strategy should maximize the expected utility over the whole exploration path, but the complexity of the problem, together with the lack of a priori information, suggests a more effective greedy approach, where the evaluation of the utility of an action is based on a single step look-ahead. However, in principle, more intelligent decisions become possible as more information is gathered about the environment.

In this paper, q^{k+1} is selected in $\mathcal{D}(q^k,k)$ so as to maximize the utility function $U(q,k) = I(q,k)$. In principle, the navigation cost from q^k to q^{k+1} could be included in U

[3]If the world model is a gridmap, this condition can be easily checked by generating a numerical 'navigation' function from $s(q^k)$ within $\mathcal{E}^k \cap B(s(q^k),\rho+R)$. Any point which can be connected to $s(q^k)$ will have a finite value of the function.

SEARCH WITH GLOBAL GROWTH

1 $\mathcal{G}^k \leftarrow$ globally expand roadmap $\mathcal{G}^{k-1}$

2 $\widehat{\mathcal{D}} \leftarrow$ extract from $\mathcal{G}^k$ the subset of configurations falling inside current admissible set $\mathcal{D}(q^k, k)$

3 $(q^{k+1}, U(q^{k+1})) \leftarrow$ find configuration in $\widehat{\mathcal{D}}$ with maximum utility

Fig. 4. A pseudocode description of the search strategy with Global Growth (GG). An instance of the search at the k-th iteration is shown.

SEARCH WITH LOCAL GROWTH

1 $\mathcal{T}^k \leftarrow$ expand tree rooted at q^k within C-space ball with center q^k and radius δ *%local search%*

2 $\widehat{\mathcal{D}} \leftarrow$ extract from $\mathcal{T}^k$ the subset of configurations falling inside admissible set $\mathcal{D}(q^k, k)$

3 $(q^{k+1}, U(q^{k+1})) \leftarrow$ find configuration in $\widehat{\mathcal{D}}$ with maximum utility

4 **if** $U(q^{k+1}) < U_{\min}$ *%global search%*

5 $\mathcal{L}^k \leftarrow$ expand 'lazy' tree rooted at q^k inside $\mathcal{D}(q^k, k)$ *%constrained expansion - no collision-checking%*

6 $\widehat{\mathcal{D}} \leftarrow$ extract from $\mathcal{L}^k$ a subset of configurations which are safe at step k and have utility $U \geq U_{\min}$

7 $(q^{k+1}, U(q^{k+1})) \leftarrow$ find a configuration in $\widehat{\mathcal{D}}$ reachable from q^k

Fig. 5. A pseudocode description of the search strategy with Local Growth (LG). An instance of the search at the k-th iteration is shown.

in order to reduce possible erratic behaviors. However, as shown in the following sections, the proposed definition of $\mathcal{D}(q^k, k)$ and an appropriate search strategy (see Sect. V-D.2) naturally limit erratic behaviors.

D. Search Strategies

During the exploration, a model of the configuration space is incrementally updated for (i) searching new view configurations and (ii) performing planning operations. Since manipulators typically have high-dimensional configuration spaces, it is convenient to use sampling based approaches to incrementally grow a roadmap which captures the connectivity of the current safe region.

In particular, let $\mathcal{G}^k$ be the roadmap built at step k in the safe region $\mathcal{S}^k$. In $\mathcal{G}^k$, a node represents a configuration that is safe at step k, while an arc between two nodes represents a local path that is safe at step k and connects the two configurations.

Once $\mathcal{E}^k$ is computed merging $\mathcal{V}(q^k)$ with $\mathcal{E}^{k-1}$, the roadmap $\mathcal{G}^k$ is obtained expanding $\mathcal{G}^{k-1}$. During this expansion process, additional sampled configurations which are safe at step k are added to $\mathcal{G}^{k-1}$. In order to find these configurations a collision checking is performed in the reconstructed world model at step k: according to this model, $\mathcal{E}^k$ is the available free world and $\partial\mathcal{E}^k$ is the obstacle boundary. Note that, in this framework, the SET built at step k represents the path actually traveled by the robot on the roadmap $\mathcal{G}^k$.

Two main instances of the SET method can be obtained depending on the strategy used for growing the roadmap. SET with Global Growth (SET-GG), which iteratively performs a global extension of the roadmap. SET with Local Growth (SET-LG), which builds the roadmap in the form of a forest of trees, each one 'locally' grown around a stored view configuration.

1) SET-GG: In this strategy the roadmap $\mathcal{G}^k$ is globally expanded using a sampling-based approach such as *(i)* a multi-query PRM algorithm or *(ii)* a single-query single-tree algorithm (RRT or EST). A pseudocode description of the strategy is shown in Fig. 4. At first, the roadmap $\mathcal{G}^{k-1}$ is globally expanded to obtain $\mathcal{G}^k$ (line 1). Here, one of the above mentioned sampling-based techniques is used. Then, the subset $\widehat{\mathcal{D}}$ of configurations falling inside the current admissible set $\mathcal{D}(q^k, k)$ is extracted from $\mathcal{G}^k$ (line 2). Note that $\widehat{\mathcal{D}}$ represents an estimate of $\mathcal{D}(q^k, k)$. At this point, q^{k+1} is found as a configuration of $\widehat{\mathcal{D}}$ with maximum utility $U(q^{k+1})$ (line 3). It is worth noting that this approach inherently performs a uniform sampling over the free configuration space.

2) SET-LG: Here, the roadmap $\mathcal{G}^k$ is built in the form of a connected forest of trees. Each of these trees is rooted at a distinct view configuration. Hence, in this case, a node of the SET represents a view configuration together with the tree rooted at that configuration. Note that, at each step, the SET-LG preliminary performs a local growth around q^k in the attempt to locally maximize the utility function, then, when no local informed configurations are found, it allows a global search (performing occasional long jumps).

A pseudocode description of the SET-LG strategy is shown in Fig. 5. At first, a tree $\mathcal{T}^k$ rooted at the current configuration q^k is expanded (line 1). Its expansion is constrained within a C-space ball with center q^k and radius δ. Here a single-query single-tree algorithm such as RRT or EST can be used. Next, the subset $\widehat{\mathcal{D}}$ of configurations falling inside the current admissible set $\mathcal{D}(q^k, k)$ is extracted from $\mathcal{T}^k$ (line 2). At this point, a candidate view configuration q^{k+1} is found as a configuration of $\widehat{\mathcal{D}}$ with maximum utility $U(q^{k+1})$ (line 3).

Note that in this first phase, the locally constrained expansion generates candidate view configurations which are distant from q^k at most δ. This mechanism automatically limits the navigation cost of the next robot motion and avoids the problematic definition of a mixed utility function (where an explicit penalty must be put on traveled distance in order to avoid problematic behaviors).

Afterwards, if $U(q^{k+1}) \geq U_{\min}$, q^{k+1} becomes the new view configuration and the search routine return. Otherwise,

a global search in the set $\mathcal{D}(q^k, k)$ is performed. This is accomplished in three steps (line 5–7). At first (line 5), a tree $\mathcal{L}^k$ rooted at q^k is expanded in the admissible set $\mathcal{D}(q^k, k)$: during this constrained expansion no collision checking is performed ('lazy' expansion). Here, RRTs are preferable for their rapid C-space exploration. Next (line 6), a subset $\widehat{\mathcal{D}}$ of configurations which are safe at step k and have utility $U \geq U_{\min}$ are extracted from $\mathcal{L}^k$. Then (line 7), a single-query planner is invoked to find a configuration of $\widehat{\mathcal{D}}$ which is reachable through a path that is safe at step k. If this planner fails to find reachable configurations, then it returns $U(q^{k+1}) = 0$. In simulations, we obtained excellent results building $\widehat{\mathcal{D}}$ just as a single configuration with maximum utility.

It is worth noting that the SET-LG mainly performs a non-uniform sampling over the free configuration space. In fact, the distinct trees rooted at the view configurations may expand in overlapping C-space regions. This unwanted result can be almost avoided by suitably selecting the radius δ of the constraining C-space balls.

E. Path Planning

Once a new view configuration q^{k+1} has been selected, the path-planner must compute a safe path connecting q^k to q^{k+1}. In the SET method, planning depends on the used search strategy.

In SET-GG, a safe path can be easily found on the roadmap $\mathcal{G}^k$. In SET-LG, two cases are possible: 1) q^{k+1} belongs to the tree $\mathcal{T}^k$ 2) q^{k+1} belongs to the lazy tree $\mathcal{L}^k$. In the first case, a safe path can be easily found on $\mathcal{T}^k$. In the second case, the adopted search strategy (Fig. 5, line 7) automatically returns a path that is safe at step k. For simplicity, this has not been made explicit in the pseudocode.

VI. Simulations

In this section we report simulation results obtained implementing the presented SET-based exploration strategies in Move3D [15], a software platform developed at LAAS-CNRS and dedicated to motion planning[4]. The proposed algorithms have been extensively tested in several scenes (both in 2D and 3D worlds) using different robot manipulators (both fixed-base and mobile manipulators). For lack of space, we report here only the results obtained in the six scenes depicted in Fig. 6. The first three refer to fixed-base planar manipulators in which the last revolute joint is used to rotate the rangefinder within a $120°$ planar cone. The last three scenes contain spatial manipulators with the last two revolute joints used to rotate the sensor within a $120° \times 120°$ spatial cone. In all the cases the rangefinder is used with a perception range $R = 1.2$ m and opening angle $\alpha = 80°$ (each robot link length approximatively ranges from 0.3 m to 0.8 m). Its linear and angular resolutions are respectively 0.01 m and $1°$. At the start of the exploration $\mathcal{E}^0$ is assumed to be known from an exogenous source (often a reasonable assumption) and is not computed through the 'bootstrap' formula (1). In particular, $\mathcal{E}^0$ is a free box containing the robot at q^0 and its size is 200% of $\mathcal{R}(q^0) \cup \mathcal{V}(q^0)$ on the average.

[4]Move3D is at the origin of the product KineoWorks currently marketed by the company Kineo CAM (www.kineocam.com).

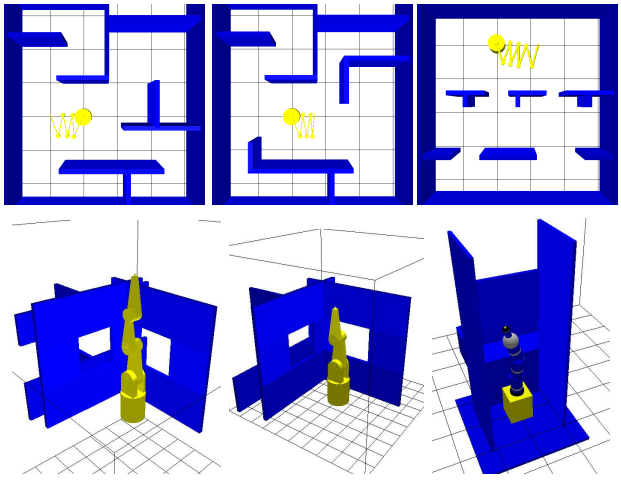

Fig. 6. *Top*: 2D scenes (from left to right): A6R, B7R, C9R. *Bottom*: 3D scenes (from left to right), D6R, E5R, F9R. In each world name, the first character identifies the environment, while the last two refer to the number of robot revolute (R) joints.

In all the simulations, we used 2D or 3D gridmaps as world model (with a 0.1m grid resolution). Quadtrees/octrees were conveniently utilized to represent (and efficiently operate on) the free boundary and the obstacle boundary. Simulations were performed on a Intel Centrino Duo 2x1.8 GHz, 2GB RAM, running Fedora Core 8.

A. Sampling Methods

As described above, the SET method uses sampling-based methods to find the next view configuration in the current admissible set $\mathcal{D}(q^k, k)$. In SET-GG, the global roadmap $\mathcal{G}^k$ is incrementally expanded using PRM or RRT.

In SET-LG, we found that the RRT method is more preferable. In particular, RRT-Extend is used for building the local tree $\mathcal{T}^k$ (Fig. 5, line 1) around the current configuration q^k, while RRT-Connect is more conveniently used for lazy tree expansions[5] (Fig. 5, line 5). Bidirectional RRT-Connect is then used as a single-query planner to find path towards the candidate configurations extracted form the lazy tree (Fig. 5, line 7).

In all these methods, kd-trees are utilized to efficiently perform nearest neighbor searches, uniform random sampling is applied, path smoothing is performed in order to 'filter' the jaggedness of the planned paths. Note that the design of an appropriate metric is critical for an efficient exploration of the configuration space [16].

B. Parameter Choice

The presented SET-based algorithms contain various parameters. For sake of clarity, we recall them below.

• *Selection Parameters*. These are ρ, the radius of the world ball which implicitly defines the admissible set, and

[5]Due to its more 'aggressive' attitude in C-space explorations.

World	*NS*	$\mathcal{W}_{\%}$	*T*	*NNGR*	*CDC*
A6R	42 (0.76%)	100.00% (0.00%)	16.2 min	52689	446314
B7R	42 (0.62%)	98.67% (0.08%)	13.4 min	40218	383995
C9R	51 (3.40%)	98.53% (0.087%)	45.86 min	14571	339952
D6R	80 (0.00%)	92.7% (0.082%)	46.2 min	45405	253504
E5R	45 (0.54%)	94.50% (0.284%)	42.4 min	32664	271735
F9R	43 (1.33%)	100.00% (0.000%)	29.8 min	34822	262885

Fig. 7. Results obtained with SET-LG: Averages values with their relative variances (in brackets).

$U_{\min} = I_{\min}$, the minimum required information gain for the next view configuration. In simulations, we obtained very good performances with $\rho \simeq R$. In all scenes, $\rho = 1000$mm and $U_{\min}$ is set equal to three times a voxel volume.

• *RRT Parameters*. Each RRT expansion is performed for a maximum number of iterations $K_{\max}$. Moreover, in SET-LG, a C-space ball with radius δ is used for constraining the preliminary local tree expansion (Fig. 5, line 1). For local and global RRT expansions we found very good results with $K_{\max} \simeq 3000$, whereas, for lazy expansions we used $K_{\max} \simeq 7000$. As for the bidirectional RRT, an even higher number of iterations was required ($K_{\max} \simeq 40000$). Typically, we set $\delta \simeq 0.1\delta_M$ where δ_M is the maximum distance between two points in the robot configuration space. This selection typically reduces the non-uniform sampling problem of SET-LG.

C. Performance Indexes

The performances of the algorithms are evaluated in terms of the following indexes.

• *Number of Scans*, denoted by *NS*. It is the total number of view configurations stored in the SET at the end of exploration.

• *World Coverage*, denoted by $\mathcal{W}_{\%}$. It represents the percentage of the free world included in the final explored region. Note that this percentage is evaluated w.r.t. to an estimate of the free world which can be explored by the robot, i.e., the set of points $p \in \mathcal{W}_{\text{free}}$ such that $p \in \mathcal{V}(q)$ for some $q \in \mathcal{C}_{\text{free}}$.

• *Simulation Time*, denoted by T. It is the overall time required to simulate a robot exploration. Note that, during simulations, many threads run at the same time (these include sensor simulation, world/robot/map visualization, path smoothing) and their computational costs affect the overall simulation time. It is worth reporting that the average simulation time significantly decrease (by 30% on the average) if only the fundamental exploration threads run during simulations (e.g., 3D visualization, path smoothing are off).

• *Collision Detection Calls*, denoted by *CDC*. It is the total number of collision detection calls performed during an exploration.

• *Number of Nodes of the Global Roadmap*, denoted by *NNGR*. It is the total number of nodes belonging to the roadmap $\mathcal{G}^k$ at the end of exploration.

D. Results

Two typical exploration processes obtained with SET-LG are depicted in Figs. 8 and 9. Note that the obstacles (in blue) are unknown to the robot. Indeed, they are incrementally reconstructed during the exploration and their known portions

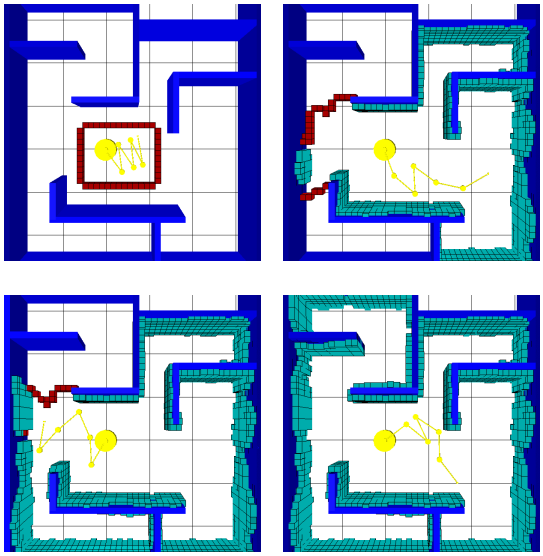

Fig. 8. Exploration process in world B7R.

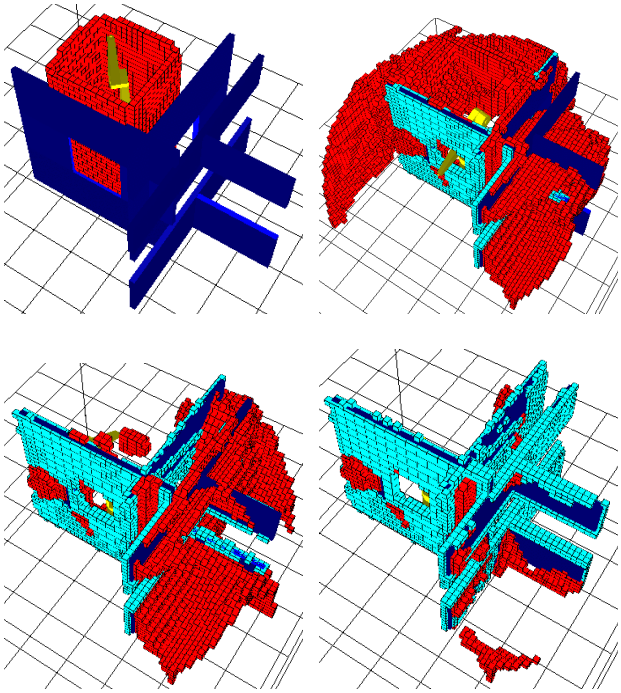

Fig. 9. Exploration process in world D6R.

are represented by the obstacle boundary $\partial\mathcal{E}^k_{\text{obs}}$ (light-blue cells). In each frame, the free boundary $\partial\mathcal{E}^k_{\text{free}}$ (red cells) is also shown. In Fig. 8 (a 2D world), the free boundary is always completely depicted and the explored region $\mathcal{E}^k$ can be clearly identified as the portion of space enclosed by $\partial\mathcal{E}^k = \partial\mathcal{E}^k_{\text{obs}} \cup \partial\mathcal{E}^k_{\text{free}}$. Conversely, in Fig. 9, only the internal portions of the free boundary (which can collide with the robot) are shown.

A typical exploration in worlds C9R and F9R is shown in the video clip attachment to the paper. Other simulations (including snake-like robots) are available at the webpage `http://www.dis.uniroma1.it/~labrob/research/SET.html`.

Figure 7 summarizes results obtained with SET-LG. In view of the randomized nature of our method, these are averaged over 20 simulation runs (relative variances are reported in brackets). Note that the average world coverage is always close to 100% in all the worlds (with an almost zero relative variance). This very good performance shows that exploration was always (almost) completed by the algorithm. Note that a complete exploration can be forced by setting $U_{\min} = 0$ (at the cost of increasing the simulation time).

E. Comparison of SET-LG with SET-GG

For lack of space, results obtained with SET-GG are not here reported. However, simulations showed that SET-LG performs much better than SET-GG, both in terms of speed of search and simulation time. In particular, we found that, for the same maximum number of iterations $K_{\max}$, the world coverage of SET-GG decreases on the average by 10% w.r.t. SET-LG, whereas simulation time increases by 40%.

These results can be justified by a comparative analysis of the two methods. At each step, the SET-method has two main computational costs: the first is due to the extension of the roadmap $\mathcal{G}^k$, while the second depends on the extraction of a subset of candidate configurations lying in $\mathcal{D}(q^k, k)$ from the roadmap.

In particular, at each step, SET-GG expands a global roadmap $\mathcal{G}^k$ which spreads uniformly over the whole safe region as k increases. Clearly, the number of nodes stored in $\mathcal{G}^k$ continuously grows. This causes a parallel continuous increment of both the computational costs described above.

On the other hand, at each step, SET-LG expands a new local tree $\mathcal{T}^k$ around the current view configuration q^k. Each of these trees, by construction, has a bounded number of nodes. Hence, with such mechanism, both the described computational costs are in principle bounded and held constant.

The local growth performed by SET-LG brings another important advantage. It inherently focuses the search process around the current view configuration q^k. This is typically convenient since, at least in the initial stages of the exploration, new informative configurations are likely to be contained in a neighbourhood of the last view configuration q^k. On the other hand, global roadmap expansion results in the problematic dispersion of new samples in uninformative configuration-space regions. Also, smaller traveled distance in C-space means less energy and smaller exploration time.

VII. Conclusion

We have presented a method for sensor-based exploration of unknown environments by a general robotic system equipped with rangefinders. The method is based on the incremental generation of a data structure called Sensor-based Exploration Tree (SET). The generation of the next action is driven by information at the world level, where the perception process takes place. In particular, the frontiers of the explored region are used to guide the search for informative view configurations. Various exploration strategies may be obtained by instantiating the general SET method with different sampling techniques. Two of these, SET-GG and SET-LG have been described and critically compared by simulations in 2D and the 3D worlds. We showed that the local growth performed by SET-LG has advantages.

We are currently working to carry out a completeness analysis of the SET algorithm. Moreover, a SET algorithm for robotic systems provided with many rangefinders is currently under testing (see `http://www.dis.uniroma1.it/~labrob/research/SET.html`). Future works will also address the inclusion of uncertainty both in the sensor model and world representation.

References

[1] Y. Yu and K. Gupta, "Sensor-based probabilistic roadmaps: experiments with an eye-in-hand system," *Advanced Robotics*, vol. 14, pp. 515–536(22).

[2] J. Ahuactzin and A. Portilla, "A basic algorithm and data structures for sensor-based path planning in unknown environments," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2000, pp. 902–908.

[3] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *1997 IEEE Int. Conf. on Robotics and Automation*, 1997, pp. 146–151.

[4] B. Yamauchi, A. Schultz, and W. Adams, "Mobile robot exploration and map-building with continuous localization," in *IEEE Int. Conf. on Robotics and Automation*, 1998, pp. 3715–3720.

[5] A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte, "An experiment in integrated exploration," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 1, 2002, pp. 534–539.

[6] H. H. Gonzalez-Banos and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *Int. J. Robotics Research*, vol. 21, no. 10, pp. 829–848, 2002.

[7] L. Freda and G. Oriolo, "Frontier-based probabilistic strategies for sensor-based exploration," in *IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 3892–3898.

[8] E. Kruse, R. Gutsche, and F. Wahl, "Efficient, iterative, sensor based 3-d map building using rating functions in configuration space," in *IEEE Int. Conf. on Robotics and Automation*, vol. 2, 1996, pp. 1067–1072.

[9] P. Renton, M. Greenspan, H. ElMaraghy, and H. Zghal, "Plan-n-scan: A robotic system for collision-free autonomous exploration and workspace mapping," *Journal of Intelligent and Robotic Systems*, vol. 24, pp. 207–234(28), 1999.

[10] M. Fernandez, K. Gupta, and J. Fraile, "Simultaneous path planning and exploration for manipulators with eye and skin sensors," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2003, pp. 914–919.

[11] Y. Yu and K. Gupta, "C-space entropy: A measure for view planning and exploration for general robot-sensor systems in unknown environments," *Int. J. Robotics Research*, vol. 23, no. 12, pp. 1197–1223, 2004.

[12] G. Oriolo, M. Vendittelli, L. Freda, and L. Troso, "The SRT method: Randomized strategies for exploration," in *IEEE Int. Conf. on Robotics and Automation*, 2004, pp. 4688–4694.

[13] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.

[14] J.-C. Latombe, *Robot Motion Planning*. Norwell, MA: Kluwer Academic Publishers, 1991.

[15] T. Simeon, J.-P. Laumond, and F. Lamiraux, "Move3d: A generic platform for path planning," in *4th Int. Symp. on Assembly and Task Planning*, 2001, pp. 25–30.

[16] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, B. R. Donald, K. M. Lynch, and D. Rus, Eds. Wellesley, MA: A K Peters, 2001, ch. 10, pp. 293–308.