2015 IEEE International Conference on Robotics and Automation (ICRA)
Washington State Convention Center
Seattle, Washington, May 26-30, 2015

Ouroboros: using potential field in unexplored regions to close loops

Vitor A. M. Jorge Renan Maffei Guilherme S. Franco Jessica Daltrozo Mariane Giambastiani
Mariana Kolberg Edson Prestes

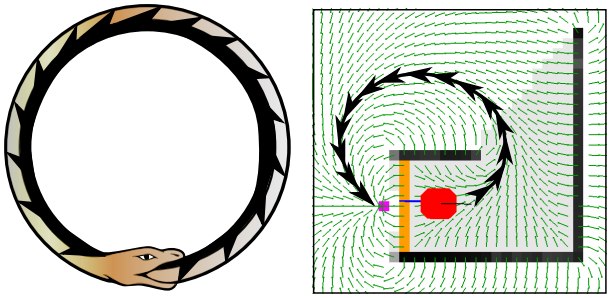
***Abstract*— An autonomous robot requires a map of the environment for many tasks. Yet, in many cases, this map is unavailable and the robot must build one in real-time, in the so-called integrated exploration task. Several integrated exploration approaches adopt some sort of loop-closing strategy combined with an online simultaneous localization and mapping (SLAM) technique. This is important because the robot can reduce the uncertainty about its pose by revisiting known areas. One solution for environment exploration is to use the vector field computed from the numeric solution of a Boundary Value Problem (BVP). This approach is called BVP Path Planner and generates smooth and free of local minima potential fields. However, this planner cannot actively close loops and does not scale well in large scenarios. In this paper we present a technique which performs active loop closure using the BVP Path Planner. Our proposal takes advantage of the potential of unexplored regions, and induces the robot to close loops by placing dynamic barriers at the visited space. The update of the potential field is boosted using a local window charged by a Voronoi diagram of the environment containing global information. We show through experimental results the effectiveness of the technique with a thorough discussion of its characteristics.**

I. Introduction

Integrated exploration (active SLAM) is fundamental for robots [1] since they rely on it to autonomously explore the environment as fast as possible without losing map accuracy. In previous works [2], [3], the numeric solution of a boundary value problem (BVP) using Laplace Equation was adapted for exploratory tasks. Its implementations is straightforward; there is no local minima in the resulting potential; and the robot moves in the direction of the closest/largest unexplored regions connecting the free space. Even though the computational cost can be high, it can be mitigated using multi-resolution BVP [4].

A robot using BVP will never go back to a previously known region, unless such region is in the path of another unknown region that is dragging the robot. In an ideal world, there would be no need to go back to any region, considering perfect observation and motion models (in static environments). However, in an integrated exploration the robot must move, localize itself, and map the environment concurrently. For that purpose, most methods rely on online Simultaneous Localization and Mapping (SLAM) strategies [5] such as the Rao-Blackwellized Particle Filter (RBPF) [6]. This implies the active use of known regions by the robot to correct localization/mapping errors. These corrections mainly occur when a distinct known region is revisited. Still, revisiting unknown regions should be done carefully, because it may lead to undesirable results. For example, by continuously revisiting a given region of the environment, the uncertainty may drastically decrease, leading to a locally correct solution. However, the remaining hypotheses may not be sufficient to provide the correct global solution [7], [8].

Institute of Informatics, Universidade Federal do Rio Grande do Sul, Porto Alegre, Brazil vamjorge, rqmaffei, gsfranco, jdbarbosa, mtgiambastiani, mariana.kolberg, prestes@inf.ufrgs.br



(a) (b)

Fig. 1. *Ouroboros* symbol (a), depicting a serpent eating its own tail. Influence of tractor point and shield in the potential field (b). In (b), we show: robot (red); obstacles (black); shield (orange); tractor point (pink); loop road (black stylized line); and the potential field (green arrows).

Besides, during the integrated exploration process, the map of the environment is built incrementally using SLAM. For this reason, the robot motion planning relies on the available partial map, and thus it cannot know in advance when a path will result in a long corridor, a vast empty space, a dead-end or a loop. This makes it difficult to choose where to go. Among the available places some are more interesting than others. A loop is one example of distinct point of interest. If the robot timely closes a loop, a revisit to a previously known region is performed and localization and mapping errors can be corrected. Still, failure to do so might result in extensive cumulative odometry errors which may not be recoverable using online SLAM techniques. The detection of loop closure opportunities is commonly handled using topological maps [7] and checking for nearby unconnected nodes in the surroundings of the robot position. It is important to control the entire loop closing process to avoid particle depletion, since the robot may lose important particle diversity by continuously revisiting the same loop.

In this paper we propose a new way to detect loops and consistently control the loop closing during the integrated exploration process using potential fields. The name of our technique, *Ouroboros* (see Fig.1(a)), comes from its intrinsic meaning – "cyclicality". The technique alternates between going towards the unexplored regions and closing loops.

978-1-4799-6923-4/15/$31.00 ©2015 IEEE

The former behavior sets attraction potentials in unexplored areas, while the latter is performed defining the pair "tractor point" (goal) and "shield" (wall blocking the direct influence of tractor point over the robot), see Fig. 1(b). A Voronoi diagram [9], [10] of the free and unexplored space is used to aid the loop detection process and boost the potential field computation – performed over the Voronoi cells and inside the local window.

This paper is presented as follows. Section II presents the related work. Section III details the *Ourobouros* strategy. Section IV defines the experimental apparatus for comparing BVP and Ouroboros. Finally, in Section V we discuss the limitations of our method and future work.

II. Related Work

Integrated exploration has been subject of several interesting works in recent years.

Makarenko et al. [11] introduce the integrated (autonomous) exploration problem and its relations with mapping, localization and path planning. They propose a navigation strategy based on a utility function which considers the map construction coverage, accuracy, and exploration speed. The localization accuracy is measured using a Kalman filter at a few selected locations during planning. The best positions are used as beacons for the robot. They present partial maps with good initial results.

Amigoni [1] performs experiments comparing different exploration strategies and shows that techniques which balance utility and cost are better than those which use only utility. Later, Amigoni et al. [12] proposed a technique that blends information gain and traveling cost to construct maps using salient points in the map as safe zones for the robot.

Stachniss et al. [7] developed an integrated exploration technique that merges a grid-based Rao-Blackwellized Particle Filter (RBPF) and an active loop closing technique, using a topological map. This strategy is only effective if the robot can see the end of the loop. In another work, Stachniss et al. [8] devised an entropy-based approach, to alternate between closing loops, re-localization and moving to unknown regions. The idea is to keep the robot inside a loop only while the diversity is not degraded. Carlone et al. [13], however, argue that calibrating the weights of linear combination of utility and cost functions requires a careful setting which can be map-dependent. Blanco et al. [5] add that this type of mathematical treatment sometimes leads to undefined values, specially after loop closing. They rely on the entropy of the expected map of the RBPF – an integration of the map hypotheses of all particles – to detect opportunities to close loops and also to guide the robot during the exploration, stopping the loop closing process when necessary.

In the area of potential fields, Prestes et al. [2] developed an exploration strategy relying on the Dirichlet boundary value problem (BVP) for the Laplace equation. Unexplored regions are given low potential (goals), while the walls are given high potential - the potential of free space is solved numerically using the finite-differences method. When there are no more unexplored frontier cells, the robot stops. The method was improved [3] by using a local window with size varying according to the smallest sonar reading. This local map is only effective if there are frontier cells inside the local window. Otherwise, the global map needs to be used to calculate the potential field. Later, Silveira et al. [4] proposed a fast multi-grid approach to solve the BVP performance problems while computing the global potential field. However, this method still needs to be tested in exploration tasks, since its parameterization may be map dependent.

Tungadi and Kleemam [14] developed an exploration technique that uses a Voronoi diagram to detect and close loops. Their loop-path strategy uses a depth-first search algorithm in the graph to check for loops that end at the desired location. The algorithm ignores small loops, trying to close the remaining ones. Vallvé and Andrade-Cetto [15] devised a decision theoretic approach for exploration which minimizes map and path entropies. The method is executed considering a gradient descent over the the potential information field, producing better maps than competing strategies. Leung et al. [16] devised an exploration which uses attractors as auxiliary information to the local planner (Model Predictive Control). The attractors push the robot towards desired locations to fulfill the active SLAM task. The method is composed of three states: improve map; exploration; and localization improvement. The idea is similar to the use of attractor points in our work. That is, we use additional features to help a local planner [2]: a shield feature and a single attractor point. Using these elements, we are able to locally generate a global loop closing behavior.

Recently [17], we have devised a strategy for exploration and patrolling of environments using BVP. We added to the BVP a new boundary condition, defined by the time of the last visit of the robot. This boundary condition, inserted on the center of the free space (i.e. the cells of the Voronoi diagram), is used to guide the robot to regions visited long time ago. The robot always has nearby attractive potentials, therefore it can locally decide where to go, allowing the potential field to be computed only in a local window. This is particularly good to keep revisiting known places, however, closer attractive potentials override the attraction of distant potentials, therefore the method may lose the capacity to fast guide the robot to distant places.

In this paper, we propose the use of the Voronoi diagram to fast propagate the potential from anywhere in the environment as well as to detect and close loops. The main difference to some of the presented techniques, is that we do not require explicit loop searches, since the potential's propagation over the Voronoi naturally attracts the robot towards the smallest loop.

III. Ouroboros explained

A. Ouroboros core: Shields & Tractor Points

In the BVP exploration [2], the robot performs the map coverage using the gradient descent of the potential field generated over a grid. This potential field is obtained by

solving, using the finite difference method, the Laplace equation

$$\nabla^2 \, p(\mathbf{r}) = 0, \tag{1}$$

where $p(\mathbf{r})$ is the potential value at position $\mathbf{r}$ in the free space. Obstacles and unknown regions are both considered as Dirichlet boundary conditions. Specifically, the potential value associated to obstacles cells is set to the maximum (1), while the potential value associated to unvisited cells is set to the minimum (0).

However, a frontier-guided behavior is not advisable as the single strategy of an integrated exploration approach. Additionally, the robot must return to visited areas to improve its localization. The idea of going back to a region coming from a different passage is what lies in the core of our approach. We transpose this idea into the BVP exploration by introducing the notion of *tractor points* and *shields* (see Fig. 1(b)).

A tractor point is an attractive potential (sink), which has to be placed in a visited area behind the robot. It must be placed over known regions, or else we could guide the robot towards out-of-reach areas. When using tractor points, the unexplored regions are no more boundary conditions, therefore the potential field is computed over both explored and unexplored regions. The shield is a virtual wall used to block the direct influence of the tractor point. It is always placed behind the robot and in front of the tractor point, as close as possible to the robot. This enforces the robot to close a loop (if exists) in order to reach the tractor point. The endpoints of a shield must be connected to obstacles, otherwise the potential field would circumvent the shield, nullifying our strategy. Fig. 2 presents different scenarios for the generation of the shield and the tractor point. As we can see, the only scenario that respects the above mentioned conditions is scenario (c).

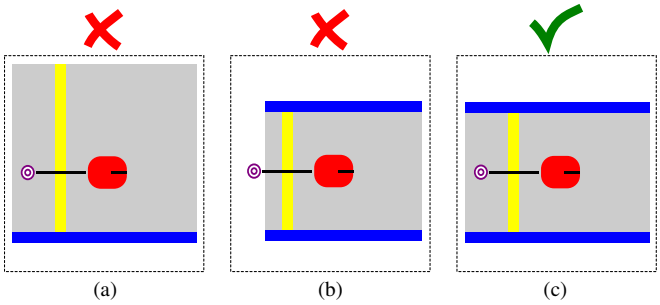


Fig. 2. Example of invalid/valid scenarios for the generation of shield and tractor point. Robot (red), shield (yellow), tractor point (purple), obstacles (blue), known region (grey).

When the robot returns to the tractor point, closing a loop in the process, both tractor point and shield are disabled, and all the cells near the path traversed by the robot during the loop closure are marked. The next tractor point must be placed in an unmarked area, therefore, each closed loop is associated to just one tractor point. Note that this does not prevent revisits as long as they are in the path to unexplored regions or to other loops.

B. Ouroboros potential field: using Voronoi diagram

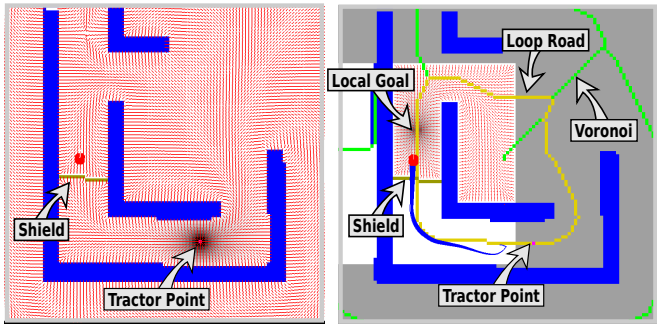
Using a grid as the map representation for the BVP exploration is an easy solution to implement, but with a major drawback regarding the update cost of the potential field. When the nearby attraction cells vanish and the available ones are distant from the robot, the entire potential field may change. In this case, the convergence time of BVP computed over grids may drastically increase.

Our approach to circumvent this problem is to avoid the propagation of the potential over the full grid. Instead, the global potential is just propagated over the center cells of the free space, i.e., the cells of a Voronoi diagram. To compute the Voronoi diagram, we perform the thinning of the free space [10], but any other Voronoi strategy which guarantees connectivity could be employed. Knowing the center cells, we compute the potential field over them, by setting the goal as the lowest potential, and the robot as the highest. In practice, since we only have punctual boundary conditions – the goal and the robot – this is equivalent to compute a wavefront propagation algorithm [18] over the center cells, starting at the goal.

With the global potential of the center cells, we can compute the potential field inside a local window surrounding the robot. We just have to set the potentials of the center cells as boundary values of the local potential field (along with the obstacles located inside the window). At this time, the computation is performed via finite difference methods, as in [2], since we have multiple boundary conditions – the center cells and the obstacles – which are not punctual. This local potential field can be used by the robot to smoothly navigate towards the goal, while avoiding obstacles. Another advantage in using a local window, is that we can set a single local goal at the cell with the lowest potential at the robot's surroundings, rather than fixing the potentials of every center cell inside the window. This not only simplifies the computation of the potential, but the local potential field becomes smoother.

Since the size of the local window is stable, the cost to update the local potential is low. However, the Voronoi inside the local window may disappear if the robot gets too far from obstacles. In this case, our solution is to grow the local window until it becomes possible to find a valid local goal. This idea is similar to Prestes et al. [3]. Still, in the worst case, the maximum size of the local window will be the maximum distance from the Voronoi to the walls.

Fig. 3 illustrates differences between the global and the local computation of the potential field. The global form is easy to implement, given that it just requires the definition of the tractor point and the shield. The implementation complexity of the latter form is higher due to the maintenance of additional structures, such as the Voronoi diagram, and the necessity of more rigor in the definitions of loops and local goals. However, computing the potential field using BVP in a large grid has severe implications in terms of cost and quality of the results, which is circumvented through the use of a local window.



(a) Global (b) Local

Fig. 3. Global vs Local computation of the potential field.

C. Ouroboros blueprint: the algorithm

Ouroboros is presented in Algorithm 1. It is initialized without shield or tractor point (line 1). The idea is to initially move the robot to the frontiers of known space, since, at the beginning, it is not possible to set the goal or build a shield in the area behind the robot (not visited yet).

Algorithm 1: *Ouroboros* Integrated Exploration

```
1  initialize(previousTractorPoint = ∅, previousShield = ∅)
2  while TRUE do
3      updateVoronoi()
4      robotCell = moveToVoronoi(robotPose)
5      if ∃ previousTractorPoint then
6          tractorPoint = moveToVoronoi(previousTractorPoint)
7          if dist(robotCell, tractorPoint) < Threshold then
8              disableTractorPoint()
9      else
10         tractorPoint = generateNewTractorPoint()
11     if ∃ tractorPoint then
12         propagatePotential(goal=tractorPoint, block=robot)
13         if ∃ previousShield then
14             loopRoad
               = buildLoopPassingByPreviousShield()
15         else
16             loopRoad = buildLoop()
17     if ∃ loopRoad then
18         shield = buildNewShield()
19     if ∃ shield then
20         propagatePotential(goal=tractorPoint, block=shield)
21     else
22         if ∃ unexploredVoronoiCells then
23             propagatePotential(goal=unexplored regions)
24         else
25             exit()
26     defineLocalGoal()
27     computeLocalPotential()
28     followPotentialField()
29     previousTractorPoint=tractorPoint;
30     previousShield=shield;
```

The first step in the main loop is to compute the Voronoi diagram with the current map of the environment (line 3). The Voronoi cells are computed through the thinning of the map (using both known and unknown cells, discounting obstacles). We start from a box slightly larger than the known area, to allow the existence of paths through unexplored regions. This continuous update is important because the Voronoi diagram changes as the known space increases. Additionally, since we use a RBPF for SLAM, the map can suffer drastic transformations in subsequent iterations, specially if the particles dispersion is large. That said, all the information placed over the Voronoi cells must be updated at each iteration, starting by the cell corresponding to the robot pose – moved to the nearest Voronoi cell (line 4).

The second step is the update of the tractor point. If there is a previous tractor point, we just move it to the current Voronoi (line 6), otherwise, we try to generate a new tractor point behind the robot (line 10). As the tractor point must be a reachable cell, we disable it if it falls over an obstacle when the map changes or if the robot reaches it, meaning that the loop was closed (line 8).

The third step is the construction of the loop road with the associated new shield. Given that the tractor point is valid, we try to build a road connecting it to the robot via two different paths – one connecting it to the robot's front and other connecting it to the robot's back. To do this, we compute the potential field over the Voronoi cells, considering the tractor point as the single goal, and the robot as a blocker (line 12). By blocking the potential propagation in the robot cell, we can easily detect the existence of a loop by checking if the potential originated in the tractor point reaches the robot from more than one direction. When this is not the case, the loop is invalid, and the robot can keep exploring the environment. Otherwise, the loop road is built by following the gradient descent of the potential field until reaching the tractor point, after departing from the Voronoi cells in the robot's 8-neighborhood.

It is worth noting that the loop road must be updated at each iteration, due to variations in the Voronoi diagram. There are two different conditions to construct the loop road: one is when the robot is already closing a loop (line 14) and other when it is not (line 16). The latter is rather simple, and corresponds to the strategy described above. The former, on the other hand, requires some extra precautions, because the resulting loop must connect the tractor point, the robot and the shield used, and stored, in the previous loop closing iteration. If these constraints are not respected, there is no guarantee that the robot will close the previously established loop. In extreme cases, the robot may end-up confined in adjacent loops. Thus, when such conditions cannot be satisfied we do not generate a loop road.

The fourth step is the update of the shield, which must be built over the loop road, orthogonally to the Voronoi. To keep the shield close to the robot, the method starts searching for a valid shield in the cells just behind the robot. The search continues until a valid shield is built or the tractor point is reached, meaning that we do not have an appropriate loop. A shield is valid when its two endpoints are connected to obstacles and when it crosses only one Voronoi branch. This is important to ensure that no path, other than the one assigned to the loop road will be blocked by the shield.

The fifth step is the propagation of the potentials, either from the tractor point considering the new shield, or from the unexplored cells. If the method is successful in all three previous steps – i.e. setting a tractor point, a loop road and a shield – it propagates the potential from the tractor point using the shield as blocker (just like the potentials propa-

gation in the loop road construction). Or else, it propagates the potentials from the Voronoi cells in the border of the free space. When there are no more reachable unexplored Voronoi cells, the method ends because the exploration of the environment is complete.

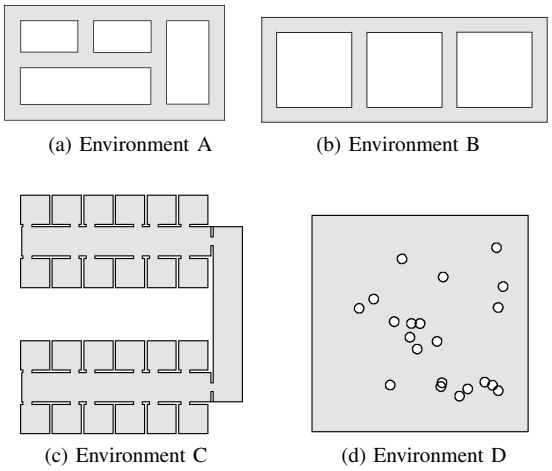
Finally, in the sixth step we define a local goal inside a local window surrounding the robot (line 26) and compute the traditional BVP inside such region (line 27), as explained in Section III-B. The local goal is always placed in the Voronoi cell with the smallest potential, i.e. the one closest to the global goal (tractor point or unexplored cells), inside the region encompassed by a circle of 2m radius around the robot. Note that the farther the local goal, the smoother is the potential field computed with BVP, therefore we must keep some distance between the local goal and the robot to avoid abrupt changes in the potential field, what could derail the navigation process. Lastly, the robot moves according to the gradient descent of the local potential field (line 28).

IV. Experiments

The evaluation of *Ouroboros* was made through the comparison with the traditional BVP exploration, in experiments using a Pioneer 3-DX mobile robot, equipped with a SICK LMS 200 laser range-finder. The experiments were performed in four simulated environments, presented in Fig. 4. Environment A contains four adjacent loops of different lengths ($44m$, $46m$, $58m$ and $78m$), while Environment B contains three adjacent loops of same length ($72m$). Environment C, of size $23m \times 25m$, is a scenario with rooms and long corridors, without loops. Environment D is a sparse square scenario, of $20m \times 20m$, containing multiple small obstacles randomly distributed in the space, simulating an unstructured environment with trees. Each experiment configuration was performed 15 times. We also performed experiments in a real scenario, whose results are presented in this section. Both methods used a regular grid with cells of $10 \times 10cm^2$. In all experiments with Ouroboros, the potentials were computed over a local square window of 61×61 cells, centered in the robot, yielding $3m$ of range. Both methods were combined with the same RBPF SLAM strategy using a laser range-finder as sensor, and an occupancy grid as the map representation [19]). Also, the same number of particles was used in all experiments: 300 particles.

Table I show the results of the experiments according to three metrics: the error (euclidean distance) between the particles positions and the real robot positions; the total time of exploration required for each method; and the final area size in comparison to the ground truth, from where we can estimate the percentage of the map left uncovered (or erroneously defined as covered). To help in the analysis of results, we present, in Fig. 5, one resulting map for each configuration. The chosen maps were the ones associated with the median error of each set of runs, in other words, for each configuration there were seven better maps and seven worse maps than the one shown here.

In a general analysis, Ouroboros presented the best results in Environments A and B – those with multiple loops.



(a) Environment A (b) Environment B

(c) Environment C (d) Environment D

Fig. 4. Environments used in the experiments, with obstacles in black and free-space in grey.

	BVP		*Ouroboros*		diff
	mean	std	mean	std	%
Position Error (m)	1.01	0.61	0.40	0.25	-60.4
Total Time (s)	4881.9	957.7	4399.3	399.7	-9.9
Final Area Size (%)	105.2	1.31	99.7	0.25	—

(a) Environment A

	BVP		*Ouroboros*		diff
	mean	std	mean	std	%
Position Error (m)	2.54	2.34	0.49	0.25	-80.5
Total Time (s)	8585.4	2796.8	5231.7	835.6	-39.1
Final Area Size (%)	141.8	39.2	100.8	0.32	—

(b) Environment B

	BVP		*Ouroboros*		diff
	mean	std	mean	std	%
Position Error (m)	0.57	0.22	0.37	0.18	-35.1
Total Time (s)	3549.7	182.0	4985.8	431.1	+40.5
Final Area Size (%)	91.1	0.57	99.9	0.36	—

(c) Environment C

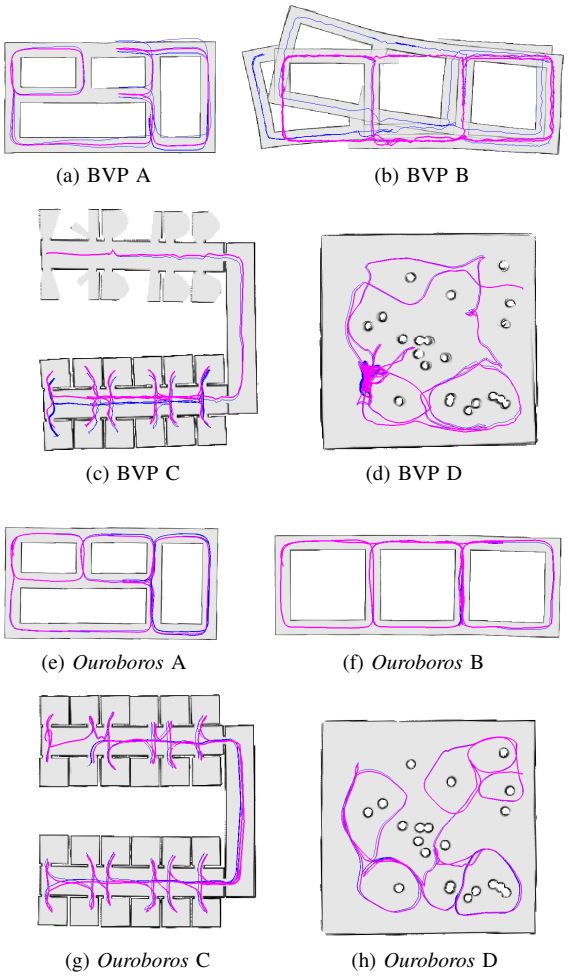
	BVP		*Ouroboros*		diff
	mean	std	mean	std	%
Position Error (m)	0.27	0.13	0.20	0.10	-25.4
Total Time (s)	5823.9	891.3	3691.3	400.4	-36.6
Final Area Size (%)	100.1	0.43	98.8	0.34	—

(d) Environment D

TABLE I
RESULTS OF THE SIMULATED EXPERIMENTS.

Environments C and D were more problematic, yet the results were also good. In C, the robot tried to close loops in numerous occasions, but it was unsuccessful given the absence of loops. Nonetheless, it could sequentially visit each room of the scenario. In D, Ouroboros had difficult to build consistent valid loops because it could not place the shield between such small obstacles. On the other hand, this avoided closing loops in cluttered areas (when the loop road sometimes pass too close to obstacles).

Comparing the performances of Ouroboros and BVP, we perceive that BVP consistently left a non negligible portion of uncovered area in Environment C, as we could see in Fig. 5(c). This is supported by the results of the final area sizes in relation to the ground-truth, as shown in Table I, where



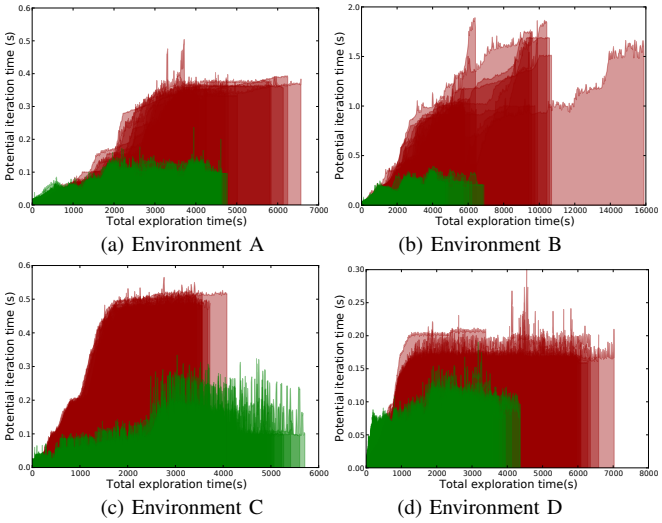
(a) BVP A (b) BVP B (c) BVP C (d) BVP D (e) *Ouroboros* A (f) *Ouroboros* B (g) *Ouroboros* C (h) *Ouroboros* D

Fig. 5. Results obtained by each method in the simulated environments, with the real robot path (pink) and the estimated path (blue).

BVP ended with almost 10% of uncovered area in such case. In Environment B, the coverage error of BVP is large due to the huge flaws in the map construction, resulting from its naive navigation strategy.

Regarding the total time of exploration, Table I shows that *Ouroboros* was generally faster than the BVP exploration. Although its focus at closing loops leads the robot to continuously revisit known regions – what could delay the exploration – such revisits tend to avoid leaving small gaps on the map, which is one of the main problems of BVP. BVP roughly covers the whole area pretty fast, but leaves openings in the map that requires further inspection. This leads to different problematic scenarios. At best, the robot must travel long distances to complete the exploration. However, it can turn into an indecisive behavior, where the goal neither is strong enough to attract the robot, nor too weak to allow the convergence of the potential field (this happens in Environment D). But it can be worse: if the attractive potentials are too small, too far from the robot, or passing through narrow passages, the potential field may suddenly flatten, compromising the exploratory process (the robot stops its navigation since there is no gradient descent to follow). Indeed, this was the cause for the premature ending of the BVP exploration in Environment C.

Fig. 6 presents plots depicting the variation of the time associated to the potential field update during the exploration process. We can see along the x axis that, with exception to Environment C, *Ouroboros* always finished sooner than BVP. In relation to the potential field update time (y axis), *Ouroboros* was also faster than BVP. The reason is that computing the potential field over the full grid becomes too costly as the size of the known area grows. In the beginning of the exploration, BVP is always faster than *Ouroboros* due to the overhead in the *Ouroboros* computation, but this changes after a few minutes of exploration. Note that, despite using a local window, the *Ouroboros* iteration time slightly varies due to the variations in the local window size and the global Voronoi computation.



(a) Environment A (b) Environment B (c) Environment C (d) Environment D

Fig. 6. Total time of exploration in seconds (x axis) and time per iteration in seconds of the potential field update (y axis) during the 15 runs in each simulated scenario, for Ouroboros (green) and BVP (red).

Regarding the error between the estimated path and the real robot path, *Ouroboros* consistently obtained better results, as we can see in Table I. Additionally, we performed the Wilcoxon rank-sum test [20] over the results to confirm that, in the tested environments, *Ouroboros* presented significant improvements in the error quality when compared to BVP (p-values of 0.00002 in A, 0.00001 in B, 0.0006 in C and 0.04 in D). Fig. 7 shows a boxplot to compare the errors concentration during the process. The error is represented in the vertical axis, highlighting the median, the quartiles and the maximum and minimum errors, discarding outliers. We can observe once again that the largest difference between *Ouroboros* and BVP occurs in the multi-loops scenarios, specially in Environment B, where the upper quartile of *Ouroboros* is below the lower quartile of BVP.

Finally, we present examples of visual results obtained with the real robot in a building floor of $72m \times 13m$ containing two adjacent loops. Ouroboros closed both loops sequentially, then it visited the front and back entries of the

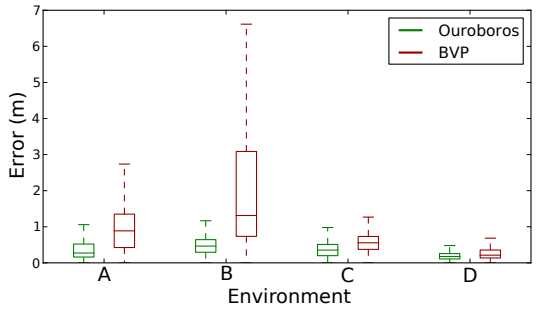
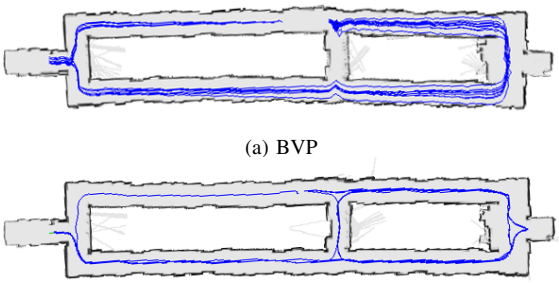


Fig. 7. Boxplot of the errors for *Ouroboros* and BVP, in all simulated environments.



(a) BVP

(b) Ouroboros

Fig. 8. Examples of results obtained in a real environment, showing the particles paths in blue.

building to complete the exploration. Contrarily, BVP did not close the two small loops, and skipped some passages, what considerably increased the uncertainty of the SLAM process. Despite this fact, both methods obtained good results. Nevertheless, the purpose of such tests were to check if ouroboros was suitable for real world applications, and the answer was positive.

V. Discussion

In this work we have proposed a new integrated exploration strategy using potential fields. Our technique has proven to be effective in four simulated scenarios when compared to the frontier-based exploration. Such scenarios included realistic structured and unstructured environments.

Our key contributions are:

(i) A loop-closure strategy using potential fields that naturally goes towards the smallest loop found, given the robot position and the knowledge it currently has about the environment;

(ii) A way to use the Voronoi diagram to charge a local window surrounding the robot. This enables the computation of the BVP locally, leading the robot to the closest unexplored regions or in the direction of the nearest possible loop closure opportunity;

There are some relevant considerations on using our strategy. We have successfully used fixed shields to explore different types of maps. Yet, if the fixed shield is placed at some region of the map that does not contain loops, the robot may be trapped into the loop-closing behavior for a while, until it detects the absence of loops. Finally, our approach does not require Voronoi edge pruning, because the robot will not walk over leaves that are known to lead it to obstacles (high potential).

We consider the concept of shields and tractor points useful for exploration and navigation. Moreover, in fully explored maps, the effect of shields and tractor points may be helpful to continuously revisit and update the map. And more, *Ouroboros* can be combined with utility metrics to avoid undesired conditions, such as staying in a long loop for long time or taking too long to detect a non existent loop. There is also the possibility to combine the effect of different shields and tractor points working together. However, all these possibilities need to be further investigated.

References

[1] F. Amigoni, "Experimental evaluation of some exploration strategies for mobile robots," in *Proc. of IEEE ICRA*, may 2008, pp. 2818–2823.

[2] E. Prestes, P. M. Engel, M. Trevisan, and M. A. P. Idiart, "Exploration method using harmonic functions," *Robotics and Autonomous Systems*, vol. 40, no. 1, pp. 25–42, 2002.

[3] E. Prestes, M. Trevisan, M. A. P. Idiart, and P. M. Engel, "Bvp-exploration: further improvements," in *Proc. of IEEE/RSJ IROS*, 2003, pp. 3239–3244.

[4] R. Silveira, E. Prestes, and L. Nedel, "Fast path planning using multi-resolution boundary value problems," in *Proc. of IEEE/RSJ IROS*, 2010, pp. 4710–4715.

[5] J. L. Blanco, J. A. F.-. Madrigal, and J. Gonzalez, "A novel measure of uncertainty for mobile robot slam with rao-blackwellized particle filters," *Int. Journal of Robotic Research*, vol. 27, pp. 73–89, 2008.

[6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series)*, Cambridge, MA, 2005.

[7] C. Stachniss, D. Hahnel, and W. Burgard, "Exploration with active loop-closing for fastslam," in *Proc. of IEEE/RSJ IROS*, vol. 2, 2004, pp. 1505–1510.

[8] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using rao-blackwellized particle filters," in *Proc. of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[9] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*, Cambridge, MA, 2005.

[10] Z. Guo and R. W. Hall, "Parallel thinning with two-subiteration algorithms," *Commun. ACM*, vol. 32, no. 3, pp. 359–373, mar 1989.

[11] A. A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte, "An experiment in integrated exploration," in *Proc. of IEEE/RSJ IROS*, 2002, pp. 534–539.

[12] F. Amigoni and V. Caglioti, "An information-based exploration strategy for environment mapping with mobile robots," *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 684–699, 2010.

[13] L. Carlone, J. Du, M. K. Ng, B. Bona, and M. Indri, "An application of kullback-leibler divergence to active slam and exploration with particle filters," in *Proc. of IEEE/RSJ IROS*, 2010, pp. 287–293.

[14] F. Tungadi and L. Kleeman, "Loop exploration for slam with fusion of advanced sonar features and laser polar scan matching," in *Proc. of IEEE/RSJ IROS*, 2009, pp. 388–394.

[15] J. Vallvé and J. Andrade-Cetto, "Potential information fields for mobile robot exploration," *Robotics and Autonomous Systems*, 2014.

[16] C. Leung, S. Huang, and G. Dissanayake, "Active slam using model predictive control and attractor based exploration," in *Proc. of IEEE/RSJ IROS*, 2006, pp. 5026–5031.

[17] R. Maffei, V. A. M. Jorge, E. Prestes, and M. Kolberg, "Integrated exploration using time-based potential rails," in *Proc. of IEEE ICRA*, 2014.

[18] S. M. LaValle, *Planning Algorithms*. Cambridge U. Press, may 2006.

[19] C. Stachniss, *Robotic Mapping and Exploration*, 1st ed., ser. Springer Tracts in Advanced Robotics. Springer, may 2009.

[20] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. pp. 80–83, 1945.