Abstract—Robots are often required to explore a terrain, thus the problem of covering the most areas of interest is recurrent and has been addressed by various algorithms. In this work, a novel algorithm to select stopping positions where sensing takes place in the presence of obstacles is proposed. The coverage problem is formulated as a Mixed Integer Linear Programming (MILP) problem and a solver is used to perform the optimization. Computer simulation is employed to demonstrate that the proposed algorithm is a satisfactory alternative to solve the coverage problem.

Keywords—Environment exploration, Coverage, Robot, MILP.

1 Introduction

Autonomous robots are often attributed tasks that involve traveling a certain terrain in order to map it, search for an item, find a target and so on. The robot is usually equipped with sensors capable of scanning a certain region around its current position. Therefore, at every position, the robot presents a so-called sensing region which it explores with its sensors. The definition of the positions the robot should occupy to perform the task is known as the coverage problem (Galceran and Carreras, 2013).

Many solutions to the coverage problem involve the division of the space in cells. Afterwards, the cells are explored based on a pre-defined strategy, such as a zig-zag walk. However, in the presence of obstacles, the coverage problem demands more complicated algorithms to define the cells. A comprehensive survey of the algorithms to perform coverage path planning is presented in (Galceran and Carreras, 2013).

In (Jeddisaravi et al., 2014) a single robot was required to search a certain terrain for an object in the presence of polytopic obstacles. In that paper, the authors assumed that the robot demanded some time to search the area around it. Thus, the problem to be solved was to find a set of positions where the robot should stop and perform the searching. The solution they proposed involved the minimization of the region of intersection between the search regions and the obstacles, to avoid occlusion of the sensors and the maximization of the sum of the distances between the stopping positions, to avoid exploring the same regions more than once, which does not result in more information. In order to do that, a generalized Voronoi diagram (GVD) was built and then the stopping positions were defined within it. Objective functions involving the occlusion area of the sensors and the distance between the stopping positions were minimized via a multi-objective genetic algorithm (GA) and the Pareto-front was determined. Afterwards, one of the solutions in the Pareto-front was selected based on weights attributed to each of the objective functions.

This problem is in principle similar to the one addressed in (Cortés et al., 2004), where multiple sensing agents must be placed in an environment so as to maximize the covered area. The difference lies in the fact that in (Cortés et al., 2004) the area must be covered at all times, but since there are multiple agents, it becomes similar to the problem of covering the area along a certain time interval with only one agent. In their paper, a density function was used to evaluate the possible profit of locating agents in specific positions at the terrain. The Voronoi regions of the agents were computed and a gradient descent method was used to perform the ensuing optimization problem. More recently, (Teraoka et al., 2011) combined a Voronoi description with the potential fields method to perform agent allocation in the presence of obstacles.

The problem of finding trajectories that maximize the covered area over a convex region in a given time interval was addressed in (Ahmadzadeh et al., 2008) for multiple Unmanned Aerial Vehicles (UAVs). To that end, the resulting optimization problem was solved with a Dynamic Programming approach.

A negotiation scheme was used in (Barrientos et al., 2011) in order to divide a certain area to be covered by multiple UAVs. Each UAV considers the reward for visiting a certain area, its distance to that area, the size of the portion of that area that exceeds the explored terrain and the overlap between explored areas. The vehicles attempt to maximize their utility by proposing possible tasks and then negotiating among themselves.

In the present work, an algorithm for optimal choice of the stopping positions in the presence of
obstacles is proposed. The free space is populated with the so-called interest points, which are points whose coverage is worthy. To reflect this, each point is attributed a reward value, which is only collected if that position is covered. Then, an optimization problem is posed so that the stopping positions are selected in order to maximize the total reward from coverage of interest points, given the constraints that the stopping positions may not lie inside the obstacles. This optimization problem presents a linear cost and linear constraints involving both real and integer-valued optimization variables, thus resulting in a Mixed Integer Linear Programming (MILP) problem. As a consequence, determination of the stopping positions can be carried out in one single step.

It is worthwhile to remark that dynamics are not considered in the present paper, therefore only the stopping positions are optimized, but no consideration is made regarding the trajectory between them. However, robot dynamics could be considered in the MILP formulation to encompass trajectory optimization. This possible extension is an attractive feature of the proposed method, but will not be addressed in the present work. In fact, in (Stratu et al., 2013), the coverage problem is addressed by solving a Mixed Integer Quadratic Programming (MIQP) problem, but the aim is to find a distribution such that all areas are covered, given that there are enough agents and their sensing range is not taken into account. In that paper, the problem is later converted to the optimal trajectory planning for a single agent by incorporating its dynamics.

The remainder of this paper is divided as follows. The proposed technique is presented in section 2. Section 3 introduces the scenario used for simulation. The simulation results are displayed in section 4. Finally, concluding remarks are given in section 5 as well as an outline of future works.

2 Proposed technique

2.1 Sensing regions and interest points

The environment exploration is to be carried out at selected stopping positions of the robot, where it activates its sensors and performs the search. The pre-defined number of stopping positions is \( N^p \) and their coordinates are \( x_i \) and \( y_i, 1 \leq i \leq N^p \).

The sensing range of the robot is modeled as a convex polytope. This should not become an issue, as the convex polytope can be adjusted to better approximate the sensing region. For instance, if the sensors are omnidirectional, circles can be arbitrarily well approximated by convex polytopes. As an example, if a square is deemed an inadequate approximation, one could then use a pentagon, an hexagon, and so on, until the requirements on the approximation of the circle are met. Thus, the sensing regions are defined by the polytope that is centered at the stopping positions of the robot. Let the sensing range be defined as:

\[
S = \{ r \in \mathbb{R}^2 | P^s r \leq q^s \},
\]

where \( P^s \) and \( q^s \) are constant matrices of appropriate dimensions defining the polytope. Then each sensing region is defined by translating this polytope to the \( i \)-th stopping position \( r_i = [x_i \ y_i]^T, 1 \leq i \leq N^p \). This can be written as:

\[
S_i = \{ r \in \mathbb{R}^2 | (P^s r - r_i) \leq q^s \}.
\]

In order to perform the exploration of the space, a number of \( N^s \) interest points \( p_j (1 \leq j \leq N^s) \) are defined in the free-space. The objective of the robot is to use the stopping positions so that the associated sensing regions contain as many interest points as possible. If these interest points are suitably distributed in the free-space and the \( a \) priori defined number of stopping positions is large enough, then the robot will be able to sense the best portion of the environment. This entails a need to differentiate between interest points that are contained in a sensing region and the others that are not. For this purpose, binary-valued variables can be used. Therefore, one binary variable \( b_{i,j}^s \) is used to decide whether it was imposed that the sensing region \( i \) must contain the interest point \( j \), i. e.

\[
b_{i,j}^s = \begin{cases} 
1, & \text{if it was imposed that } p_j \in S_i, \\
0, & \text{otherwise} 
\end{cases}
\]

The so-called “big-M” method (Agarwal et al., 2010) uses binary variables in conjunction with the definition of a certain constant large enough to relax the constraints in (2) for all admissible values of \( r_i \), therefore the name “big-M”. Thus imposing that the interest points lie inside the sensing regions is carried out by:

\[
P^s(p_j - r_i) \leq q^s + M^s (1 - b_{i,j}^s) \mathbf{1}_n
\]

where \( n \) is the number of rows of \( P^s \), \( M^s \in \mathbb{R}_+ \) is the big-M, i. e., the constant large enough to relax the constraints and \( \mathbf{1}_o = [1 \ 1 \ldots \ 1]^T \). Therefore, when \( b_{i,j}^s = 1 \), the constraint is imposed, otherwise it is relaxed.

2.2 Obstacle avoidance

One issue is the presence of obstacles. In this work, the \( N^o \) obstacles are represented as convex polytopes:

\[
O_k = \{ r \in \mathbb{R}^2 | P^o_k r \leq q^o_k \}, 1 \leq k \leq N^o
\]

where \( P^o_k \) and \( q^o_k \) are constant matrices of appropriate dimensions defining the polytopic obstacles.
It is required that the robot avoids them, i.e., \( r_i \notin O_k \), 1 \( \leq i \leq N^s \), 1 \( \leq k \leq N^o \). By not placing the interest points inside the obstacles, one naturally steers the stopping positions away from them. However, this does not ensure collision avoidance. Therefore, additional constraints must be imposed on the stopping positions. These are not-convex constraints, since imposing that stopping position \( r_i \) does not lie inside obstacle \( O_k \) is the same as imposing that it lies on the complement of \( O_k \), herein designated as \( \bar{O}_k \):

\[
    r_i \in \bar{O}_k \iff \exists l \in \{1, 2, \ldots, N^f\} | P^c_{k,l} r_i > q^c_{k,l}, \tag{6}
\]

with \( N^f \) being the number of sides of the obstacles (it is assumed, without loss of generality, that all obstacles have the same number of sides) and, \( P^c_{k,l} \) and \( q^c_{k,l} \), denoting the \( l \)-th rows of \( P^c_k \) and \( q^c_k \), respectively.

Again, binary-valued variables can be used in conjunction with the big-M method to impose the not-convex constraints in (6). However, an improvement can be used by assigning each of the regions defined in (6) to the particular value of a tuple \( \lambda \) of binary variables, according to (Prodan et al., 2012). Mappings \( \beta_{k,l} : \{0, 1\}^{N^b} \mapsto \mathbb{N} \) of tuples of binary variables \( \lambda \in \{0, 1\}^{N^b} \), where \( N^b = \lceil \log_2 N^f \rceil \), and \( \lambda = (b^0_{i,k,1}, b^0_{i,k,2}, \ldots, b^0_{i,k,N^b}) \), are defined as:

\[
    \beta_{k,l}(\lambda) = \begin{cases} 
    0, & \text{if } \lambda = \lambda^l \\
    L > 1, & \text{otherwise} 
    \end{cases} \tag{7}
\]

Therefore, by associating a particular value \( \lambda^l \) of a tuple to each inequality in (6), one can impose that (6) is verified. The inequalities are rewritten as

\[
    -P^c_{k,l} r_i < -q^c_{k,l} + M^o \beta_{k,l}(\lambda_{i,k}), \tag{8}
\]

and a tuple \( \lambda_{i,k} \) is associated with the \( i \)-th stopping position and the \( k \)-th obstacle. If \( \lambda_{i,k} = \lambda^l_{i,k} \), then \( \beta_{k,l}(\lambda_{i,k}) = 0 \) and the inequality in (8) becomes \( P^c_{k,l} r_i > q^c_{k,l} \). On the other hand, if \( \lambda_{i,k} \neq \lambda^l_{i,k} \), then the constraint (8) is relaxed, given that the constant \( M^o \in \mathbb{R}_+ \) is large enough.

This procedure is made possible by a linear mapping scheme, thus allowing the encoding as a MILP problem. The linear mapping can be constructed as in (Prodan et al., 2012):

\[
    \beta_{k,l}(\lambda) = a^k_{0,l} + \sum_{m=1}^{N^b} a^k_{m,l} \lambda^m_{i,k,m}, \tag{9}
\]

with the constant coefficients \( a^k_{m,l} \) given as:

\[
    a^k_{0,l} = \sum_{m=1}^{N^b} \lambda^1_{i,k,m}, \\
    a^k_{m,l} = \begin{cases} 
    1, & \text{if } \lambda^m_{i,k,m} = 0 \\
    -1, & \text{if } \lambda^m_{i,k,m} = 1 
    \end{cases} \quad 1 \leq m \leq N^b \tag{10}
\]

where \( \lambda^m_{i,k,m} \) denotes the \( m \)-th bit of the tuple \( \lambda^l_{i,k} \) associated to the \( l \)-th side of the \( k \)-th obstacle.

All that is left is to render the tuples not-associated to any complement in (6) infeasible. This can be done by using the encoding scheme of (Afonso and Galvão, 2014) along with the following inequality:

\[
    \sum_{m=1}^{N^b} 2^{m-1} b^o_{i,k,m} \leq N^f - 0.5 \tag{11}
\]

This way, each feasible tuple is associated to the complement of one of the half-planes that define the obstacle, which means that, for some \( l \in [1, N^f] \), \( \beta_{k,l}(\lambda^l_{i,k}) = 0 \) is true, thus \( P^c_{k,l} r_i > q^c_{k,l} \) is imposed.

### 2.3 Cost function

The proposed cost function involves the weighted sum of two terms. The first is related to the maximal distance between the interest points and the border of the sensing region that contains it. It is aimed at stopping the robot closer to the interest points. This is done by adding a slack variable \( \xi_i \geq 0 \) to the left-hand side in (2):

\[
    S_i = \{ r \in \mathbb{R}^2 | P^s (p_j - r) + \xi_i 1_n \leq q^s \}, \tag{12}
\]

Maximizing \( \xi_i \) results in the largest value by which the borders of \( i \)-th sensing region can be approximated towards the center \( r_i \) without ceasing to contain one of the interest points.

The second term implements a reward scheme which accounts for each interest point that is inside at least one of the sensing regions. This is carried out using the binary variables in (3) and the constraints in (4), subject to the following additional constraint:

\[
    \sum_{j=1}^{N^s} b^i_{s,j} \leq 1, \quad 1 \leq i \leq N^p. \tag{13}
\]

The inequality in (13) imposes that each interest point may be considered to lie inside none or at most one sensing region. It is important to remark that this does not mean that an interest point cannot be contained in more than one sensing region. It simply ensures that, during the optimization, the interest points are considered to be observed by only one sensing region. In turn, this ensures that the number of interest points that are inside at least one sensing regions can be determined as

\[
    \sum_{i=1}^{N^p} \sum_{j=1}^{N^s} b^i_{s,j}. \tag{14}
\]

With these considerations, the cost function is formulated as:

\[
    J = - \sum_{i=1}^{N^p} \left( \gamma_1 \xi_i + \gamma_2 \sum_{j=1}^{N^s} b^i_{s,j} \right), \tag{15}
\]
where $\gamma_1, \gamma_2 \in \mathbb{R}^+$ are constant weights chosen by the operator. The $'-'$ signal in (15) is introduced because the optimization problem is defined as a minimization.

### 2.4 Resulting optimization problem

Given the considerations above, the optimization problem can be posed as:

$$
\min_{x, y, \xi, \lambda, b_{i,j}, b_{i,k,m}} J
$$

subject to:

1. $r_i = [x_i \ y_i]^T$, $1 \leq i \leq N^p$.
2. $r_i \in T$, $1 \leq i \leq N^p$.
3. $\lambda_{i,k} = (b_{i,k,1}^p, b_{i,k,2}^p, \ldots, b_{i,k,N^s})$.
4. $P_s^p (p_j - r_i) + \xi_j \leq q^s + M^s (1 - b_{i,j}^p)$.
5. $1 \leq i \leq N^p$, $1 \leq j \leq N^s$.
6. $\sum_{j=1}^{N^s} b_{i,j}^p \leq 1$, $1 \leq i \leq N^p$.
7. $-\xi_j \leq 0$, $1 \leq i \leq N^p$.
8. $-P_{s}^p i r_j \leq -q_{i,j}^s + M^o \beta_{i,j}(\lambda_{i,k}) + \epsilon$.
9. $1 \leq i \leq N^p$, $1 \leq k \leq N^o$, $1 \leq l \leq N^f$.
10. $\sum_{m=1}^{N^o} 2^{m-1} b_{i,k,m} \leq N^f - 0.5$.
11. $1 \leq i \leq N^p$, $1 \leq k \leq N^o$.

where $T$ is the polytopic set of allowed positions, i.e., the terrain to be explored, and $\epsilon > 0$ is an arbitrarily small constant used to transform the $'\leq'$ to $'<'$, removing the border of the obstacles from the free-space.

The number of variables and constraints involved in the resulting optimization problem is summarized in Table 1. As can be seen, the number of sensing positions $N^p$ has the most decisive role in the computational complexity of the optimization problem.

<table>
<thead>
<tr>
<th>Obstacle #</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$3 \leq x \leq 15$  $27 \leq y \leq 40$</td>
</tr>
<tr>
<td>2</td>
<td>$22 \leq x \leq 30$  $15 \leq y \leq 25$</td>
</tr>
<tr>
<td>3</td>
<td>$20 \leq x \leq 28$  $38 \leq y \leq 46$</td>
</tr>
<tr>
<td>4</td>
<td>$10 \leq x \leq 16$  $10 \leq y \leq 20$</td>
</tr>
</tbody>
</table>

A grid of equally-spaced points were taken on the terrain, totaling 6 points on the $x$ direction and 10 points on the $y$ direction. Interest points intersecting the obstacles were removed. The scenario is depicted in Fig. 1, where the black rectangles represent the obstacles and the black points are the interest points.

### 3 Simulation scenario

This section presents the simulation scenario that is used to illustrate the employment of the proposed technique.

The terrain to be explored is defined by the following inequalities: $0 \leq x \leq 30$ and $0 \leq y \leq 50$. Four rectangular obstacles are considered. The associated constraints are presented in Table 2.

The sensing range was approximated by squares with sides of 8 units of length and a total of $N^p = 13$ sensing positions were selected. The weights in the cost function were set to $\gamma_1 = \ldots = \gamma_2 = \ldots$.
8.36 \times 10^{-5} \quad \text{and} \quad \gamma_2 = 10. \quad \text{These were chosen so that the emphasis on minimizing the distance of the interest points to the border of the sensing region does not exceed the primary objective of maximizing the number of observed interest points.}

A personal computer equipped with an Intel® i7-4790 processor with 3.6GHz clock and 8GB of RAM was used for the simulations. The CPLEX toolbox from IBM ILOG was used for solution of the MILP problem, in Matlab environment.

4 Results and discussion

In this section, the results for the stopping positions selection according to problem (16) applied to the scenario in Section 3 are presented. In Fig. 2, the stopping positions and the associated square sensing regions resulting from the solution of the optimization problem are depicted as ‘×’ and gray squares, respectively. As can be seen, each square covers a number of interest points (from two up to four, in this example). Their centers do not intersect the obstacles nor do they lie outside the explored terrain.

The optimal cost was −440.0011. The integer part of the cost is due to the coverage of the interest points by the sensing regions. As for the fractional part, it is related to the minimal distance of the interest points to the side of the sensing region that covers them. The introduction of this term in the cost function can be seen to induce a desirable feature of keeping the interest points as far from the border of the sensing regions as possible. With a suitable choice of the weight of the terms in the cost function, the sensing is not jeopardized by the maximization of this minimal distance. The coverage of the space is seen to be uniform, exploring the entire environment and few overlaps of sensing squares occurred.

It is interesting to notice that the proposed optimization problem has many equivalent solutions, as two different sensing squares that visit the same points, for instance, result in the same overall cost. For that reason, the MILP solver has to explore a number of equivalent nodes. Table 3 shows the cost versus the limit that was imposed on the maximal number of nodes to be explored before returning the feasible solution which presented the least cost. As can be seen, with a small number of 10 nodes, the cost is around −420. It requires an increase to 5 \times 10^3 nodes to drop the cost to approximately −430. Then, 5 \times 10^4 nodes are required to reduce the cost to around −440. This cost is maintained for more nodes, up to 5 \times 10^5. From that number of nodes on, the computational load becomes excessive and the problem could not be solved in reasonable time with the employed computer. Thus, one may terminate with 5 \times 10^4 nodes, as was done in the results presented in Fig. 2, if the produced solution is acceptable bearing in mind that it may not be the optimal solution.

The times taken to solve the problem with different number of nodes are also depicted in Table 3. It is important to emphasize that the proposed technique is aimed at planning stopping positions offline. Therefore, the times taken to plan the stopping positions must be seen as part of a trade off between optimality and the time the operator disposes of to deploy the robot. However, large planning times are in principle not prohibitive in this case.

Table 3: Cost and times for solution versus maximal number of nodes.

<table>
<thead>
<tr>
<th># of nodes</th>
<th>Cost</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>−420.0010</td>
<td>1.6</td>
</tr>
<tr>
<td>50</td>
<td>−420.0010</td>
<td>1.7</td>
</tr>
<tr>
<td>100</td>
<td>−420.0010</td>
<td>1.7</td>
</tr>
<tr>
<td>500</td>
<td>−420.0010</td>
<td>1.9</td>
</tr>
<tr>
<td>1000</td>
<td>−420.0010</td>
<td>2.5</td>
</tr>
<tr>
<td>5000</td>
<td>−430.0011</td>
<td>19.0</td>
</tr>
<tr>
<td>1 \times 10^4</td>
<td>−430.0011</td>
<td>32.1</td>
</tr>
<tr>
<td>5 \times 10^4</td>
<td>−440.0011</td>
<td>142.1</td>
</tr>
<tr>
<td>1 \times 10^5</td>
<td>−440.0011</td>
<td>279.7</td>
</tr>
<tr>
<td>5 \times 10^5</td>
<td>−440.0011</td>
<td>1595.3</td>
</tr>
</tbody>
</table>

Figure 2: Interest points and sensing squares approximating the sensing range.
5 Conclusion

The proposed algorithm successfully solved the coverage problem, delivering the solution in a reasonable amount of time, while avoiding obstacles. It was noted that there are many solutions presenting the same overall cost, which can hinder the minimization. To circumvent this, a suitable limit may be imposed on the number of nodes that are explored before returning a solution.

Future works could encompass the use of additional decision variables to allow a variable number of stopping positions. Moreover, robot dynamics could be considered in the optimization problem in order to search for the optimal coverage trajectory.

Acknowledgments

The authors acknowledge the support of FAPESP (grants 2011/17610-0 and 2015/16240-6 and doctorate scholarship 2011/18632-8) and CNPq (research fellowship).

References


