A Robot on-line Area Coverage Approach based on the Probabilistic Lloyd Method

Murillo Rehder Batista and Rodrigo Calvo and Roseli Ap. Francelin Romero

Abstract—Area Coverage is a standard problem in which Robotics techniques can be applied. An approach to solve this problem is through techniques based on Centroidal Voronoi Tessellations (CVT), considering that each robot is a generator used to build Voronoi polygons. In this work, a new approach named by Sample Lloyd Area Coverage System (SLACS), is proposed that does not need of the explicit building of the diagram based in the Probabilistic Lloyd method to estimate a Voronoi polygon’s centroid. In addition, it is proposed a method to close Voronoi diagrams to apply in a classic Lloyd CVT procedure. Both approaches are compared in empty and room-like environments done in simulated tests using both Player interface and Stage simulator. Results obtained show that the proposed approach is well suited to solve the area coverage problem via mobile sensor deployment and it is a simple and effective substitute to a Lloyd CVT method.

Area coverage is a commonly approached robotics problems [1], in which a team of robots does coverage of an unknown environment in a way that sensor coverage is maximized while having enough proximity to maintain connection among the robots.

Many techniques in literature deal with this task. Howard and Mataric presented the use of Potential Fields to control the robots in an unknown environment, considering that robots have only local sensing capabilities. This approach makes obstacles exert repelling forces which are inverse to the distance between the robot and such obstacle. The robots halt when a equilibrium status is reached in the system. An effective use of this method demands, however, a proper tuning to navigation constants [2].

Voronoi tessellations show many interesting aspects on solving many problems inside Computer Science area, most notably in computer graphics, pattern recognition, robotics, among others [3].

In [5], it was presented a solution for area coverage problem using a team of robots based on Centroidal Voronoi Tessellations (CVTs), showing a high level of scalability and robustness. CVTs are Voronoi Tessellations where a point is both the generator and the centroid of a Voronoi polygon. Such structure can be obtained directing a generator of the Voronoi tessellation to the centroid, repeating this procedure as a cycle until Voronoi tessellation stays stable. This approximation process is known as Lloyd’s procedure for computing CVTs[6]. An approach in which Voronoi tessellations manages the robots’ distribution and Potential Fields are used to avoid obstacles was presented in [4]. The robots use the density of Voronoi polygons, considered inversely proportional to its area, to direct themselves in regard to their neighbors. A work, that used a Voronoi tessellation-based procedure was adapted to ensure good results in non-convex environments, was presented in [11]. However, these methods need a Voronoi tessellation building procedure to obtain areas or centroids and a method to close infinite Voronoi regions in order to make the centroid procedure feasible.

A probabilistic approach to compute Voronoi polygon centroids in which there is no need to explicitly build Voronoi tessellations is shown in [7], [8], named as Probabilistic Lloyd or the Lloyd-MacQueen method. Such approach was later used to manage a team of phototropic robots [9] in a way that robots reach the light source and avoid collision among themselves. But, this approach has no procedure to deal with obstacles other than the neighboring robots themselves.

Considering the area coverage problem, a new method for simple adaptation of the probabilistic approach is proposed using the proximity of the sensors available to a team of robots, named here by Sample Lloyd Area Coverage System (SLACS). This method is compared to a system that uses the classic CVT-based approach to solve the area coverage problem via mobile sensor deployment, named Truncated Cell Lloyd Area Coverage System (TCLACS). Experiments are carried out in an environment without obstacles and in an environment divided by obstacles. Simulations are performed by using the Stage simulation environment.

This paper is organized such as it follows. In Section I, CVTs are briefly explained. In Section III, TCLACS method is described. The proposed method for area covering, SLACS, is presented in Section IV. In Section V, simulation results are reported. The main contributions and relevant aspects of this paper as well as expectations for future works are highlighted in Section VI.

I. CENTROIDAL VORONOI TESSELLATIONS

Voronoi tessellations are polygonal structures built through generator samples \( x_i, i = 1..n \) in a given space \( \Omega \). The polygons respect a constraint, shown in Equation 1, which makes that each polygon contains in its space \( V_i \), a single generator sample \( x_i \), that is the nearest generator from any sample obtained from \( V_i \).

\[
V_i = \{x \in \Omega \mid \|x - x_i\| < \|x - x_j\|, \text{ for } j = 1, \cdots, n \text{ and } j \neq i\}
\]
A Voronoi tessellation can be classified as a Centroidal Voronoi Tessellation (CVT) when its generators are also centroids of the associated Voronoi polygons according to a density function \( \rho(x) \). In this work, the distance functions are assumed to be Euclidean distances and that \( \rho(x) \) is a constant. CVTs have a remarkable optimization propriety which can be related to the concepts such as variance and cost \([8]\). Figure 1 shows a typical Voronoi tessellation in contrast to a CVT. It is important to notice that to build the image a limit bound under the interval \([0, 10]\) was adopted.

The classic method to build a CVT is the Lloyd algorithm \([6]\). A step of this technique consists of building a Voronoi tessellation from a set of generator points belonging to \( \Omega \) and then directing the generator samples towards the resulting centroids. This procedure can be repeated until a convergence criterion is reached. This procedure has a behavior strikingly similar to k-means algorithm, widely known in machine learning research area.

A. The probabilistic Lloyd procedure

An alternative approach to obtain CVTs was presented in \([7]\), combining the Lloyd method and a random sampling procedure, named the MacQueen method, which had the property of obtaining CVTs without explicit calculation of Voronoi tessellations but needed much more (despite faster) iterations to achieve the same convergence criteria. This method was considered a generalized algorithm which could be interpreted as a probabilistic Lloyd procedure or a generalization of MacQueen method. Algorithm 1 shows its steps. Constants \( \alpha_1, \alpha_2, \beta_1 \) and \( \beta_2 \) control the step size of the generators towards the centroids. If \( \alpha_1 = \beta_1 = 0 \), the method is a probabilistic version of Lloyd’s algorithm.

Algorithm 1 The probabilistic Lloyd - Generalized MacQueen method

Require: Number of random samples \( s \), an initial set of generators \( z_{k,i} \), \( \alpha, \beta \), constants \( \alpha_1, \alpha_2, \beta_1, \beta_2 \) given \( \alpha_1 + \alpha_2 = 1, \beta_1 + \beta_2 = 1, \alpha_2, \beta_2 > 0 \).

1: while Convergence criteria have not been reached do
2:    Generate \( s \) random samples according to density function \( \rho(x) \);
3:    for \( i = 1 \ldots k \) do
4:        Obtain the average \( u_i \) of all random samples which
5:        are nearest to \( z_i \);
6:        \( z_i = \frac{(\alpha_1 u_i + \beta_1) z_i + (\alpha_2 u_i + \beta_2) u_i}{\alpha_1 + \beta_1 + \alpha_2 + \beta_2} \);
7:    end for
8:    end while
9: return CVT Generators \( z_{k,i} \).

II. SENSOR AND ROBOT MODELS

The robots used in TCLACS and SLACS methods are equipped by sensorial fields: proximity (or obstacle distance) sensor and an antenna.

The direct communication is provided by an antenna. It emits and detects information (or messages) around the robot with communication radius \( R_C > 0 \). The information consists of the identification (msg_id) of robots detected in the communication area and it is emitted continuously as long as the robot navigates. Moreover, the robots are equipped with a proximity sensor. It allows the robots to define their coverage area and to avoid collisions in risk situations where robots are very close to an obstacle. Due to the dimensions of the robots adopted, it is possible that some collisions occur. The coverage area of the proximity sensor is equivalent to the circumference of a circle of radius \( R_D \), \( 0 < R_D < R_C \), around the robot. The number of sensor readings is denoted by \( S, S > 0 \).

It worthy to emphasize that the trajectories generated by TCLACS and SLACS do not lead the robots to collision situations. Obstacle avoidance is an emergent behavior mechanism described here. However, there are some exceptions when a robot collides against another robot or against an obstacle according to its physical characteristics. In this sense, a mechanism for obstacle avoidance based on fuzzy logic that adjusts the movement direction of the robot using the information about its distance to an obstacle is used. It is enough to say that this mechanism is active only when the distance between the robot and an obstacle is smaller than a predefined constant \( \eta > 0 \).

In order to follow the description of the methods, consider the assumption there are \( N \) identical mobile robots \( r_k, k = 1, \ldots, N, N \in \mathbb{N} \) that move in a planar space \( Q \subset \mathbb{R}^2 \) and that an arbitrary point in \( Q \) is denoted by \( q \).
The dispersion behavior of the robots was also promoted by a mechanism based on Voronoi tessellation and Delaunay triangulation in previous works. Through principles of CVT-based methods, TCLACS builds Voronoi partitions and leads the robots towards their respective centroids.

First, a robot $r_k$ is able to detect its neighbors through messages received from antenna. This information is organized in a set of neighbors’ identifiers, $D_A = \{(msg_i,j)|0 \leq i \leq n\}$, where $n$ is the number of detected neighbors, that is, robots whose distances to a neighbor are lower than $R_C$. Thus, the position of each robot can be estimated. Therefore, the set of neighbors of robot $r_k$ is denoted by $N^k = \{(msg_i,j)|0 \leq i \leq n_k\}$, where $n_k$ is the number of neighbors of robot $r_k$.

Within neighbors $N^k$ of robot $r_k$, only those directly connected are considered to build a Voronoi partition. For this purpose, Delaunay triangulation is applied resulting in the graph $G_k(V_k,E_k)$, where $V_k$ is the set of vertices that corresponds to the neighbors’ position, including position of robot $r_k$ ($V_k = N^k \cup \{msg_i,j,k\}$), and $E_k$ is the set of edges. Neighbors directly connected to robot $r_k$ are denoted as adjacent neighbors and modeled by set $N^k_{ADJ} = \{(msg_i,j_i,msg_q,j_k) | (msg_i,j_i,msg_q,j_k) \in E_k\}$.

In Figure 3 is illustrated a scenario in which robot $r_3$ detects seven neighbors in its communication area, hence, $N^3 = \{r_1,r_2,r_4,r_5,r_6,r_7,r_8\}$. However, there are only four adjacent neighbours, $N^3_{ADJ} = \{r_1,r_2,r_4,r_7\}$.

The adjacent neighbors of robot $r_k$ are the elements of interest for building its Voronoi partition. To this end, the generators are the robot’s position $(q_k)$ and the positions of its adjacent neighbors, denoted by set $P = \{q_k\} \cup \{msg_i,j_i\}, \forall (msg_i,j_i) \in N^k_{ADJ}$. Considering the planar space $Q$ and $M^k \subset Q$ the communication area of the follower, $r_k$, at instant $t$, the Voronoi cell $V^k(P,t)$ is defined by:

$$V^k(P,t) = \{q \in L^k | \|q - q_k\| \leq \|q - msg_i,j_i\|, \forall msg_i,j_i\}$$

Occasionally, the robot can be in an open Voronoi cell. This cell is truncated by a Voronoi circle $O(q_k,R_V)$ with center in $q_k$ and radius $R_V < R_C$, such that the new truncated Voronoi cell $V^k_{RV}(P,t)$ is determined according to Equation 3. In Figure 4 is shown the instance in which an open cell is truncated.

$$V^k_{RV}(P,t) = V^k(P,t) \cap M^k_t \cap O(q_k,R_V) \quad (3)$$

To promote group dispersion reducing collision risks, it is desired that the robots stay inside to their own Voronoi cells. In this sense, the robots should be attracted to the interior of their cells while the group moves. For this reason, the centroid $C_{V,k}$, placed at $q_c$, of Voronoi cell of robot $r_k$ is computed. Hence, a centroid attraction force, $F^k_C$, is applied. As a robot moves to its centroid, the intensity of $F^k_C$ is weakened. When follower robot reaches its centroid, the intensity of $F^k_C$ is null. Here, the likelihood to collision is also null. The intensity $I^k_c$ is directly proportional to distance $d_{k,c}$ between follower robot $r_k$ and its centroid, defined by:

$$I^k_c = e^{-(1-d_{k,c})} \quad (4)$$

Therefore, the force $F^k_C$ is obtained as:

$$F^k_C = \frac{q_c - q_k}{dist_{k,c}}, I^k_c \quad (5)$$

IV. SLACS METHOD

In this the proposed method, SLACS is described. Initially, each robot through its sensors’ readings generate the samples that are used to estimate the centroid of its Voronoi partition. The generation of these samples, unlike Rounds’s approach [9], does not use any random sampling or density function. They are created through a deterministic rule from the direction of the sensors, being equally spaced between themselves. According to this approach, obstacles are treated through discarding samples created where the sensors detect obstacles. Algorithm 2 shows how the samples are generated.

After generating its samples, a robot must locate its nearest neighboring within communication range. If a neighbor $r_j$ is detected, the robot obtains the neighbor’s relative position, $q_j$, and its samples. Samples which stand nearer to robot’s neighbor are discarded and those remaining are used in Equation 6 to obtain the desired centroid $C_T$. If there is no available neighbor, the robot only uses its own samples. In Figure 5 is shown how the samples are arranged when the robots achieve their centroid position and a Voronoi partition generated from all centroids at this instant.

$$CT_r = \frac{\sum_{i=1}^{s} S_i}{s} \quad (6)$$

Fig. 3. Neighborhood of a robot defined by Delaunay triangulation: (a) neighbors detected by antenna; (b) adjacent neighbors.

Fig. 4. Truncation process of a Voronoi cell.
Algorithm 2 SLACS Sample Generation Procedure

Require: Number of directions of a sensor attached to a robot $m$, maximum sensor reach $R_D$, maximum number of samples that can be generated from a sensor direction $s_i$, a set of sensor distance readings $sr_i m_{i=1}^m$, sensor reading’s relative angle $a_i m_{i=1}^m$ to its robot.

1: for $i=1...m$ do
2: \hspace{1cm} $j = 1$;
3: \hspace{1cm} DistInterval := $R_D/s_i$;
4: \hspace{1cm} while DistInterval $\neq sr_i$ do
5: \hspace{1.5cm} $S_{i+j} = \left[ \cos(a_i)*\text{DistInterval}, \sin(a_i)*\text{DistInterval} \right]$;
6: \hspace{1cm} DistInterval := DistInterval + $R_D/s_i$;
7: \hspace{1cm} $j = j + 1$;
8: \hspace{1cm} end while
9: end for
10: return Set of samples $S_{i+j}, i=1...m, j=1...s$

Different from the original approach in which the probabilistic Lloyd’s method is proposed, there is no need to adjust a step size towards the centroid in robot area coverage, as this step is actually implicit in the robot’s own movement procedure.

Notice that SLACS can be trivially adapted to use data obtained from more than one neighboring robot by simply using their positions and points and checking which points were nearer to the robot. However, a simple neighbor configuration was adopted for the sake of simplicity and to avoid calculation of a larger number of samples. As it is shown further in the Section V, the system is able to achieve convergence despite considering a single neighbor. Also, SLACS can handle different density functions with ease by using them to generate the samples. They can manage intervals between samples coming from the same reading direction. This SLACS’s simplicity shows a significant flexibility potential.

V. EXPERIMENTAL RESULTS

Several experiment simulations are performed to evaluate preliminarily the proposed coordination strategy and to provide a comparison with the traditional Lloyd method.

The Player/Stage platform [10] is used to perform the experiments. Player operates in a client/server environment and the communication between them occurs through TCP/IP protocol. Stage is a simulator for robots and sensors in two-dimensional environments. Player/Stage models various robots and sensors simulating their exact dynamics. The robot Pioneer 2DX is chosen to be modeled. This robot is equipped with a laser range-finder able to scan the environment. Such experiments were developed using Player interface combined with Stage simulation tool.

To verify the robots dispersion and maximization of covered area, 20 robots were positioned closely to each other in a wide environment without obstacles, that is, an empty environment (90 meters high $\times$ 60 meters width). It was considered also an environment with obstacles (40 meters high $\times$ 20 meters width) and an environment with reduced dimensions. In Figure 6 is shown the initial configuration (initial position of the robots). Particularly, the small regions resultant from the division by obstacles in the environment of Figures 6(b) and 6(c) are called here as rooms. For the third environment, some passages between two rooms were blocked in order to consider a smaller environment.

It is worthy to say that the robots’ velocity, $v$, is constant. Even if the robots reach their respective centroid position in both methods, they continue moving. Another relevant fact to be emphasized is the nonholonomic characteristic of the robots. Hence, if the centroid position is in the opposite direction of the robots’ movement direction, some more steps are needed to reach the centroid. For this reason, it is possible that the robots lose communication. To overcome this constraint, the robots must stop when the distance to their closest neighbor is higher than a predetermined constant $l$, denoted as distance threshold. The nearest neighbor distance was chosen because it presents a higher collision risk. For practical purposes, Table I lists the parameters adopted.

In the first testing scenario, an empty environment is considered. Both methods are executed according to the
TABLE I
PARAMETERS ADOPTED

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_C$</td>
<td>8 m</td>
<td>communication radius</td>
</tr>
<tr>
<td>$R_D$</td>
<td>0.5$R_C$</td>
<td>detection radius</td>
</tr>
<tr>
<td>$v$</td>
<td>0.2 m/s</td>
<td>robots' velocity</td>
</tr>
<tr>
<td>$l$</td>
<td>0.85$R_C$</td>
<td>distance threshold</td>
</tr>
<tr>
<td>$m$</td>
<td>10</td>
<td>direction of sensor</td>
</tr>
<tr>
<td>$s$</td>
<td>5</td>
<td>samples for direction sensor</td>
</tr>
<tr>
<td>$\eta$</td>
<td>0.2 m</td>
<td>obstacle avoidance threshold</td>
</tr>
</tbody>
</table>

In both methods, the robots were able to spread out, increasing the coverage area. The percentage of the covered area is presented in Table II. It can be observed that covered area in TCLACS is slightly higher than SLACS. Also, the gaps among the robots in SLACS method cause a reduction of the covered area overlap. However, in this case, the likelihood to the robots loss the connection (or communication) among them is higher. Since TCLACS method considers all adjacent neighbors to build a Voronoi cell, the centroid position is set such that a robot is moved in order to maintain a constant (or near) distance to them. In a simpler way, SLACS method counts on only one neighbor to obtain the centroid position. Thus, while a robot drives away from its closest neighbor, it can get closer to other. Although the dispersion is not uniform, the covered area length is similar to that one in TCLACS method.

Additional data is shown in Figures 8 and 9. There is a comparison between SLACS and TCLACS methods, concerning the average distance of the nearest neighbor of all robots as long as they navigate in the empty environment. The Figure 8 suggests that the robots dispersion in SLACS method had intensity and effectiveness similar to TCLACS method. Moreover, it is noticed that the period when both the methods reached the equilibrium state is roughly the same (iteration 100). The equilibrium state is also observed in the Figure 9, where the standard deviation of the average distances is showed. For both methods, there is some oscillation of standard deviation in the first iterations of the simulation. It is caused due to dispersion movement performed by the robots. After a period, the robots reach their respective centroid position. In this case, the intensity of the centroid attraction force is (or nearly) null, which interrupts the robots dispersion. As it was verified that the robots reach the equilibrium state nearly at iteration 100, the standard deviation of average distances is constant after this iteration.

SLACS method had intensity and effectiveness similar to TCLACS method. Moreover, it is noticed that the period when both the methods reached the equilibrium state is roughly the same (iteration 100). The equilibrium state is also observed in the Figure 9, where the standard deviation of the average distances is showed. For both methods, there is some oscillation of standard deviation in the first iterations of the simulation. It is caused due to dispersion movement performed by the robots. After a period, the robots reach their respective centroid position. In this case, the intensity of the centroid attraction force is (or nearly) null, which interrupts the robots dispersion. As it was verified that the robots reach the equilibrium state nearly at iteration 100, the standard deviation of average distances is constant after this iteration.

![Fig. 7. Final configuration of dispersion process: a) TCLACS; b) SLACS methods.](image)

![Fig. 8. Average of the closest neighbor distances (m).](image)

![Fig. 9. Standard deviation of the average of the closest neighbor distances (m).](image)
TABLE III
COVERED AREA IN THE ENVIRONMENT WITH OBSTACLES

<table>
<thead>
<tr>
<th>Mechanism</th>
<th>Covered area (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCLACS</td>
<td>70.43</td>
</tr>
<tr>
<td>SLACS</td>
<td>72.71</td>
</tr>
</tbody>
</table>

For the next experiments, coverage methods are applied in the environment of Figure 6(b). Analogously to previous experiments, the performance of both methods is similar. Coverage area resultant from dispersion process and the percentage of the covered area in each methods are presented in Figure 10 and Table III, respectively.

As it was mentioned, one of fundamental properties that differs TCLACS and SLACS methods is the fact that TCLACS method considers all adjacent neighbors of a robot to compute its centroid position, while in SCLACS method, it is needed only one adjacent neighbor (the closest one) to obtain the centroid position. For this reason, a robot in SLACS method moves towards its closest neighbor, distancing itself from others. This characteristic might increase the likelihood an robot disconnect from the group. To avoid the partition of a group, it is needed that the closest neighbor of a robot be changed continuously. Other factor that influences the group partition is the presence of obstacles among robots. It means that it is not possible a robot obtain a neighbor’s coordinates if an obstacle interferes the communication signal. This behavior is observed in the performance of the robots dispersion when SLASC method is considered. There is the connection loss of three robots (at top left region of the environment) from the rest of the group, as it can be seen in Figure 10(b).

According to Figure 10, it can be seen that the dispersion behavior occurred with more intensity through SLACS method in comparison to the TCLACS method. That is, more rooms were covered when SLACS method is considered. However, using this method, more gaps among the robots appeared, resulting in some uncovered regions in areas surrounded by the robots. In addition, the robots using TCLACS method stay closer to each other (Figure 13(a)), covering all portions of the rooms. This lack of dispersion in TCLACS and the number of gaps in SLACS can be explained by having a map too big to cover the entire environment. Due to the area overlap in TCLACS, the percentage of area covered is slightly higher using the SLACS method.

The performance of both methods is also evaluated observing the comparison between the averages of the closest neighbor distances. Figure 11 compares these distances. It is verified that the robots in TCLACS method reach the equilibrium state at around iteration 600 when the average distance floats over the distance of 4.5 meters. For SLACS method, the robots take more time to establish the equilibrium. It is expected that the equilibrium is reached at iteration 980. The same behavior is perceived in the standard deviation of the average distances (Figure 12). In this case, the variation of the standard deviation for TCLACS method was between 1 and 1.5 meters from iteration 600. Thus, it is possible to affirm that the distance between a robot and its closest neighbor suffered a low oscillation. In SLACS method, the behavior observed in the curve of the standard deviation is possibly justified by a robot which had lost communication with its team, which results in high values for the standard deviation.

Finally, the third experiment was designed to verify the behavior of the robots in a small environment shown in Figure 6(c). In this case, the robots movement are restricted in three rooms, where it is impossible to reach the others. By using a reduced environment with the same number of robots considered in the previous experiment, it is expect that dispersion of robots be also observed, avoiding collisions.

Even in a small environment, the robots are able to make a dispersion behavior. It can be observed in Figure 13 that the robots were spread out in all rooms, preserving the connection among them. Due to the reduced dimensions of the environment, the robots tend to maintain a small distance among their neighbors. As a consequence, the number of robots in each rooms is higher than the number noticed in previous scenarios, where all passages are opened.

Since the robots are in a smaller environment, the total coverage area (considering all robots) at equilibrium state instant is satisfactory. The percentage of the covered area for both methods is similar and summarized in Table IV.
is noticed that the dispersion behavior to cover an area is not damaged for a reduced environment.

Although the rooms are crowded, TCLACS and SLACS allow robots to maximize the covered area. Moreover, the good performance of these methods causes a free-collision navigation. Analogous to previous experiments, the performance of both methods in the reduced environment is evaluated according to average of closest neighbor distances and the deviation standard obtained from these distances. The data are visualized in Figures 14 and 15, respectively.

For both methods, the average of the closest neighbor distances present the same behavior (Figure 14). Because the environment is reduced (proportionally with the number of robots) the distance among robots is smaller. Nevertheless, the distance is increased gradually as the robots are spread out. It is worth to emphasize that there were no instants where a robot was near a situation of disconnecton from the group. The values of the average distances when the robots’ dispersion is stabilized in both methods are similar, oscillating in 3.5 meters at iteration 800.

The analysis of the behavior of the average distances’ standard deviation shows a similar result between the methods. While the values of the standard deviation for TCLACS method obtain stability between a small interval, floating around 1 meter (since iteration 800), the values interval that the standard deviation assumes when the robots reach the equilibrium state is higher in SLACS method. However, in general, the standard deviation in SLACS is smaller during most of the simulation. In addition, the robots operating under SLACS were able to balance a little better the number of robots in each room again.

If convenient, $R_{C}$ value can be interpreted as a safe distance for robot communication, in which distances can

<table>
<thead>
<tr>
<th>Mechanism</th>
<th>Covered area (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCLACS</td>
<td>76.43</td>
</tr>
<tr>
<td>SLACS</td>
<td>74.72</td>
</tr>
</tbody>
</table>
be detected through signal strength analysis. Also, it is not common in literature to apply CVT-based approaches for area coverage in environments with a complexity similar to those used here, with obstacles separating rooms from each other. There might be a difficulty in CVTs on maintaining communication in such cases.

Although results show that overall SLACS obtains performance similar to TCLACS, with the exception of the quality of the communication network built, it must be noticed that not only SLACS uses a much simpler procedure, but it is using a single robot as a neighbor, obtaining a noticeable scalability advantage. Also, SLACS can handle different density functions with ease by using them to generate the samples. They can manage intervals between samples coming from the same reading direction. Furthermore, the SLACS’s simplicity shows signals of flexibility.

In both scenarios, the methods provide to the robots to travel in the environment effectively, making good area coverage considering the sensor range in comparison to environment size. In the empty environment’s case, the gaps among sensor readings were significantly larger in the proposed approach compared with the classic Lloyd method. Also, the classic approach positioned the robots in an effective manner to make connections between robots. This problem can be reduced if the distance limit from a neighbor until a robot stops is smaller. There might be, however, cases where a configuration similar to the one shown using the classic method.

The environment with rooms shows that for both cases some communication groups are formed instead of a full network consisting of all robots like in the first map. This happens because of a connection loss due to a wall between neighboring robots or because the nearest neighbor (probabilistic Lloyd based approach) or the neighbors chosen through Delaunay Triangulation (classic Lloyd approach) change during navigation. Although the coverage percentages were similar, the probabilistic Lloyd based approach achieved better results on spreading robots into other places of the environment, obtaining a more balanced robot distribution in the environment.

VI. Conclusions

In this work, a new area coverage method named SLACS was presented. This method is based on a sampling-based procedure to calculate the centroids of the polygons of a Centroidal Voronoi Tessellation called the probabilistic Lloyd method, which discards the necessity of building Voronoi polygons. Also, it offers an effective built-in reactive obstacle avoidance procedure. SLACS generates samples deterministically according to the available proximity sensor, allowing easy obstacle management and offering flexibility to the robot system. Also, a technique to close infinite neighbors despite using just a single neighbor. Also, the robots’ distribution in an environment where the number of robots is enough was more balanced in the experiment with SLACS processing algorithm. The disadvantages presented are given by the lack of Delaunay triangulations and their advantages in establishing effective hop networks. A remarkable merit of this work is in the scenarios chosen to test the area coverage: area coverage techniques inspired by Centroidal Voronoi Tessellations are rarely presented with tests in environments, where many walls are existent. As future works, some communication issues due to the adoption of using only the nearest neighbor will be identified as well as the performance of SLACS with various density functions and a proof of convergence of this method.

References