2022 IEEE 18th International Conference on Automation
Science and Engineering (CASE)
August 20-24, 2022. Mexico City, Mexico

Minimalist Coverage and Energy-Aware Tour Planning for a Mobile Robot

Anirban Ghosh, Ayan Dutta, Brian Sotolongo

***Abstract*— We study a coverage and tour planning problem wherein a robot with limited sensor coverage is assigned to serve n known points in an environment. A location is served if it is within the visibility range of the robot's sensor. However, the robot is equipped with a battery that powers the robot to travel a maximum distance of B. Several charging stations are placed in the environment so that the robot can charge itself (if needed) to complete the mission. The objective is to compute an energy-constrained tour that starts at a given start location, visits a minimum set of service locations for serving the n points of interest, and returns to the start location. We also aim to minimize the distance traveled between any two consecutive service locations in the tour via a subset of charging stations. This problem has applications in search-and-rescue and surveillance missions where such coverage and energy-aware path planning are of utmost importance. We propose a new algorithm for this problem and show its efficacy using experiments with up to 1000 points of interest on a plane. The running time of our algorithm for such a scenario was a negligible 5.33 sec.**

I. Introduction

In a real-world surveillance mission, human operators move to a certain surveillance location, search nearby for unexpected changes or targets, and move to the next service location. The service locations might be decided based on accessibility to the nearby regions or in an ad-hoc fashion. In order to accomplish the same task using a mobile robot with a given set of points of interest (POIs), the objective of the robot is to plan a tour starting from location s, moving through a set of service locations $\mathcal{L}$ similar to the human operator, before coming back to s again [3], [14]. Similar applications can be found in search and rescue missions and precision agriculture [20], [29]. However, planning such a tour while traveling the shortest distance via the service locations exactly is known by the name Traveling Salesman Problem (TSP) and it is proved to be NP-hard [18], [15].

Often, robots have a visibility/sensor range $r > 0$, and therefore, it does not need to visit every POI for providing service. Thus, $\mathcal{L}$ can be a set of locations different from the set of POIs. Finding such a set $\mathcal{L}$ that covers all POIs is not straightforward since the environment is continuous and consequently, there are infinite candidate locations to choose from. Further, in order to minimize the total time taken to complete the mission, $\mathcal{L}$ should have the minimum size. Note that this is different from sensor-based coverage [1], [23] as our objective is not to cover all the points in the environment with the sensor footprint. Instead, we are interested in covering the POI set only. On the other hand, in real-world missions, the robot might have an onboard power source such as a battery, which needs to be recharged after the robot travels B distance. Therefore, if two consecutive locations u, v in the above-mentioned surveillance tour have a longer distance than B, the robot will not be able to move to v from u. To avoid this, we assume that there are k charging stations available in the environment, which the robot can move to and get fully recharged instantaneously. However, in this setting, the plan of traveling in a straight line from point u to v becomes infeasible.

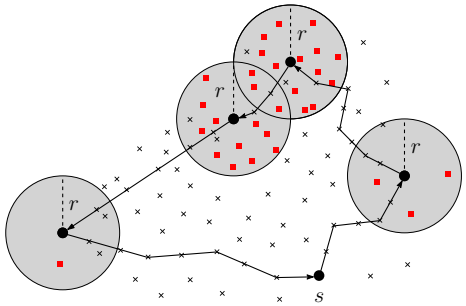


Fig. 1. In this figure, s denotes the robot's start location. A gray disk represents the area of visibility of the robot when it is located at the disk center. The points of interest (POIs) are represented using red solid squares. In this case, the set of four disk centers can be used to serve all POIs inside the environment. The cross marks represent the charging stations. A desired energy-constrained tour is shown that starts at s, visits the four disk centers, and comes back to s via a subset of the charging stations such that the distance between any two successive disk centers (service locations) is minimized. Note that two disks may intersect and multiple POIs may belong to such an intersection.

To this end, we propose a technique that 1) finds the minimum number of unit disks on the plane such that all the POIs are covered, 2) finds a TSP tour covering these disk centers while the start and the endpoint of the tour being s, and finally, 3) finds an energy-aware path for the robot to travel between any two consecutive locations in the TSP tour such that it never runs out of energy. An illustration of this is presented in Fig. 1. Our energy-aware tour planning algorithm always finds an optimal energy-aware path between any two consecutive points on the tour. We have implemented the proposed algorithms in environments consisting of up to 1000 POIs. Results show that our coverage algorithm is extremely fast – taking less than 0.3 microseconds for such a 1000-POI set. On the other hand, the energy-aware tour component of the proposed technique is understandably slower – the maximum time to compute such a tour in the

The authors are with the School of Computing at the University of North Florida, USA. {anirban.ghosh, a.dutta,n01060290}@unf.edu.
A. Ghosh was partially supported by the NSF award CCF-1947887.

same environment is 5.33 seconds.

The primary contributions of our paper are as follows:

- To the best of our knowledge, this is the first study to utilize the UNIT DISK COVER problem to cover a set of POIs on a plane while minimizing the number of service locations that the robot has to travel to.
- We have proposed an energy-aware version of the TSP routing problem such that the robot never runs out of energy while traveling from one TSP point to the next.
- Numerical results show that our proposed technique is highly scalable – taking about 5 seconds to plan an energy-aware tour while covering 1000 points of interest.

II. RELATED WORK

Coverage path planning and tour planning are popular research topics in robotics [11], [19], [31], and there is a large body of literature available. Therefore, in this section, we only focus on the studies that are the most relevant to this paper. In [30], the authors propose a technique for coverage path planning with an energy-constrained robot. Their proposed algorithm guarantees constant-factor approximation. Unlike traditional coverage planning, we are interested in covering a set of points on the plane, not necessarily all possible points, assuming the robot can serve/cover all the points in its vicinity. This is similar to the sensor-based coverage problem in robotics [1]. For example, in [23], the authors have used Generalized Voronoi diagram-based graph modeling to guarantee complete coverage. We are interested in minimizing such coverage points in the sensor-based coverage problem and our points of interest might not be scattered throughout the environment unlike such coverage [1]. TSP is a popular choice in robotics for tour planning [2], [4], [25], [27]. A survey of various routing problems studied in robotics can be found in [19]. On the other hand, a survey on TSP and its variants can be found in [15]. Euclidean TSP with neighborhood (TSPN) on disks is a generalization of the classic TSP where the points are generalized to disks (possibly overlapping). Refer to [7], [8] for approximation algorithms for TSPN on disks. Although this problem may sound similar to our setup but is different since, in our case, we need to find the disks (corresponding to the service locations) and then use their centers in energy-efficient path-planning. Although strictly not routing, recently, Sotolongo et al. [24] have proposed an energy-constrained path planning technique, where the robot's goal is to move from a start to a goal location while not violating the given energy constraint. However, we cannot apply this technique directly in this paper as we study a routing problem and we only have charging stations as the nodes in our underlying graph apart from the start and goal, but not any other waypoints. Our studied problem also has similarities with the Electric Vehicle Routing Problem (EVRP) [21] but that does not consider minimal disk coverage like ours.

Two of the closest studies to our work in this paper are [22], [26]. In [22], the authors have proposed an optimal algorithm for the coverage path planning problem under energy constraint. The main differences with our paper are 1) the authors have considered the problem of visiting all the locations on a plane whereas we consider visiting only a subset $\mathcal{L}$, 2) the environment is discretized whereas our algorithm works for a continuous plane, and finally, 3) there is assumed to be a single charging station placed in the environment, but in our paper, we can have $k \geq 1$ such stations. On the other hand, in [26], the authors have proposed a technique for a similar energy-aware tour problem. However, their proposed technique 1) does not consider the robot's visibility radius r, and therefore, assumes the service locations to be readily available unlike ours, and 2) their algorithm could scale only up to 25 service locations and 5 charging stations whereas ours scale up to more than 400 service locations and 175 charging stations.

III. PROBLEM SETUP

Let S be a set of n fixed points of interest, $p_1, p_2, \ldots, p_n$, located inside a plane environment E, needing service from a moving point robot equipped with a sensor that can serve all the points of interest which are within a pre-specified radius of r from its current location inside E. This implies, if a set $S' \subseteq S$ of points of interest are located within r distance units from the current location of the robot, it does not need to travel to the locations in S' in order to provide the desired service. Clearly, it is quite possible that a point of interest may get more than one opportunity to obtain service from the robot. Refer to Fig. 1 for an illustration.

The robot must start at a specified location $s \in E \backslash S$, serve the n locations in S, and come back to the start location, minimizing the overall distance traveled. Further, the robot is powered by a battery that helps it to travel a distance of at most B, after a single full charge. Thus, a set C of k fixed charging stations have been placed inside E to help it carry out the task of serving the n target locations. Note that in our case it is possible that $S \cap C = \emptyset$. The robot can use these charging stations as many times as it needs to, during its service tour. In this work, we assume that a single charging takes a negligible amount of time and is thus not considered in our computations.

The objective is to compute a set $\mathcal{L}$ of a minimum number of serving locations (not necessarily the same as the given point of interests) inside E and a permutation of these locations that the robot should visit starting and ending at s in order to serve the n point of interests $p_1, p_2, \ldots, p_n$, minimizing the overall distance traveled using a battery that powers it to travel a distance of at most B after a single charge. Note that it is possible that $\mathcal{L} \cap S = \emptyset$. The tour to be computed consists of locations in $\mathcal{L} \cup \{s\}$. In the cases, where the distance between two successive locations in the tour is larger than B, a set of nearby charging stations needs to be added to the tour as well. Note that $|\mathcal{L}| \leq n$ since in the worst-case if any two POIs are more than $2r$ distance units away, the robot needs to use n service locations to serve the n POIs.

When the distance between two successive service locations is much larger than B, the robot may need to charge

itself several times before it can reach the next destination. In doing so, the robot may deviate from the line segment joining the two consecutive service locations since the charging stations are spread throughout the environment E. In this regard, we consider a practical optimizing constraint: for every two consecutive serving locations in the solution, minimize the total distance traveled by the robot via a subset of the charging stations.

We define the objective of the energy-aware tour part of the problem as follows: find a tour T_e^* that starts and ends at s, serves all the locations in $\mathcal{L}$ exactly once, does not violate the energy constraint, and minimize the travel cost. The travel cost of a tour is calculated by summing up the Euclidean distances of its edges (an edge represents the straight-line path between two successive locations). Throughout this paper, the subscript e is used to denote energy awareness.

IV. Algorithms

The algorithm section is broken down into three subsections as follows. In Section IV-A, we show how to compute $\mathcal{L}$ for serving the given n points of interest. Next, we show how to compute an energy-constrained tour that starts at s, visits every point in $\mathcal{L}$, and ends at s; see Section IV-B. Finally, in Section IV-C, we present an algorithm that returns a feasible solution for the energy-aware problem.

A. Finding $\mathcal{L}$ using a covering algorithm

The geometric covering is a well-researched family of problems in computational geometry where the objective is to cover a given set X of geometric objects using the minimum number of congruent copies of some fixed shape Q. In our problem, $X = S$ and Q is a closed disk of radius r (visibility radius of the robot under consideration) since a robot located at a disk center can serve all the points of interest inside the disk having radius r. This means to minimize the size of $\mathcal{L}$, we need to find the minimum number of disks whose union covers S. This problem is popularly known by the name Unit Disk Cover problem and is being researched for over three decades; refer to [6], [10], [13] for a literature overview of this problem.

In the Unit Disk Cover problem, r is assumed to be a unit (as the name suggests) but the algorithms for it can be easily scaled depending on the value of r used. The problem is shown to be NP-Hard by Fowler et al. [9] back in 1981 and consequently, a long series of approximation algorithm has been designed having varied running times and approximation, see [6], [13]. A fast practical algorithm is proposed by Ghosh, Hicks, and Shevchenko [13], that gives 2.7-approximation[1] in practice and runs in $O(n)$ time on average where $n = |S|$. Using rigorous experiments, the authors have showed that their algorithm is up to 61.63 times faster than some of the well-known algorithms for the covering problem. Since path-planning computations are expected to run fast in practice, we adapt this algorithm to

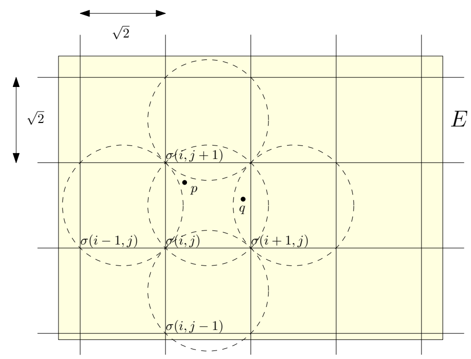


Fig. 2. An illustration of the algorithm Cover-POIs. The environment is denoted by E. In this figure, r is assumed to be a unit. The size of the used square grid is $\sqrt{2}$, as required by the algorithm. A point $p \in \sigma(i,j)$ can be covered using the unit disk $D(i,j)$ that circumscribes $\sigma(i,j)$. Note that depending on the location of p, it can also be covered by some disk in $D(i-1,j), D(i+1,j), D(i,j-1), D(i,j+1)$. Thus, before placing $D(i,j)$ to cover p, we check if any one of the four disks is already covering p and is placed previously to cover some other point of interest in S. For instance, the point of interest p can be covered by the disk $D(i,j)$ only and consequently, we are forced to use $D(i,j)$. However, the point of interest q belongs to $D(i,j) \cap D(i+1,j)$. Thus, when q is considered, first we check if $D(i+1,j)$ is already placed previously for covering some other point of interest. If yes, no action is necessary for q since it is already covered, otherwise, we place the disk $D(i,j)$ to cover q.

generate the set of $\mathcal{L}$. In the following we briefly describe the algorithm.

We overlay a $\sqrt{2}$-sized grid Ξ on the environment E; see Fig. 2. This will split E into $\sqrt{2}$-sized square cells as shown in the figure. We use $\sigma(i,j)$ to denote a cell in Ξ, where $i, j \in \mathbf{Z}$. The cell $\sigma(i,j)$ is the intersection of the four half-planes: $x \geq \sqrt{2}i, x < \sqrt{2}(i+1), y \geq \sqrt{2}j, y < \sqrt{2}(j+1)$. For every cell $\sigma(i,j) \in \Xi$, there exists a unit disk $D(i,j)$ that circumscribes $\sigma(i,j)$. We say that $D(i,j)$ is the *grid-disk* of $\sigma(i,j)$.

The points in S are considered sequentially. Let $p = (p_x, p_y)$ be the point currently being considered. Since we are using a $\sqrt{2}$-sized grid, p belongs to the cell $\sigma(i,j)$, where $i = \lfloor p_x/\sqrt{2} \rfloor$, $j = \lfloor p_y/\sqrt{2} \rfloor$. First, we check if p is covered by some previously placed grid-disk in $\{D(i-1,j), D(i+1,j), D(i,j-1), D(i,j+1)\}$. If not, then we use the disk $D(i,j)$ that is centered at the center of the cell $\sigma(i,j)$ to which the point belongs.

The centers of the grid-disks placed by this algorithm can be used as the set $\mathcal{L}$ of service locations for covering the n points of interest. Refer to Algorithm 1 for a pseudo-code.

B. Finding a tour on $\mathcal{L} \cup \{s\}$

Once the set of service locations $\mathcal{L}$ is computed using Algorithm 1, the next task is to find a minimum-weight cycle that starts at s, visits every location in $\mathcal{L}$ exactly once, and comes back to s (a tour on $\mathcal{L} \cup \{s\}$). This is a folklore problem in computing and is known by the name Metric Traveling Salesman Problem (Metric TSP). In this version of TSP, the input graph G is complete and is constructed on an input point set. Further, the edge weights satisfy the triangle inequality: for any three edges $(u,v), (v,w), (u,w)$ in G, $\texttt{weight}((u,v)) + \texttt{weight}((v,w)) > \texttt{weight}((u,w))$. In

[1]In this case, the approximation factor implies that in the worst-case, 2.7 times the optimal number of disks may be placed by this algorithm.

Algorithm 1: COVER-POIS(S)

DISKS-PLACED $= \emptyset$;
for $p \in S$ **do**
  Let $i = \lfloor p_x/\sqrt{2} \rfloor$, $j = \lfloor p_y/\sqrt{2} \rfloor$, where p_x, p_y are the x and y-coordinates of p, respectively;
  if $D(i,j) \in$ DISKS-PLACED or there is a disk $D \in \{D(i-1,j), D(i+1,j), D(i,j-1), D(i,j+1)\} \cap$ DISK-PLACED such that D covers p **then**
    continue;
  else
    DISKS-PLACED = DISKS-PLACED $\cup D(i,j)$;
  end if
end for
$\mathcal{L} = \emptyset$;
for every disk $D \in$ DISKS-PLACED **do**
  $\mathcal{L} = \mathcal{L} \cup \{(x,y)\}$ where (x,y) denotes the center of D;
end for
return $\mathcal{L}$;

our case, the input graph G is a complete graph on $\mathcal{L} \cup \{s\}$ and the weight of an edge (u,v) is the Euclidean distance between the two locations $u, v \in \mathcal{L} \cup \{s\}$. Just like the coverage problem, this is also NP-Hard [18]. In our case, we use the famous Christofides algorithm [5], [28] that gives 1.5-approximation[2] and runs in $O(|\mathcal{L}|^3)$ time. For the Metric TSP problem, this polynomial-time algorithm gives the best guarantee on the tour length till date. This plays a very important role in path planning scenarios for optimizing the time taken in completing missions. However, recently, a slightly improved randomized algorithm has been devised in [17], that gives $1.5 - \epsilon$-approximation, for some $\epsilon > 10^{-36}$. We did not use this algorithm in our work since to our knowledge, no experimental study is conducted on this algorithm till date. See [12] for an experimental study on Metric TSP. We present a pseudo-code of the Christofides algorithm in Algorithm 2.

Algorithm 2: COMPUTE-TOUR($\mathcal{L} \cup \{s\}$)

Let G be the complete graph on $\mathcal{L} \cup \{s\}$;
Compute a minimum spanning tree M for G;
Let W be the set of vertices in G that have odd degree in M and H be the subgraph of G induced by the vertices in W;
Compute a minimum-cost perfect matching P in H;
Combine the two graphs M and P to create a graph G'. However, if an edge e belongs to both M and P, create two copies of e in G';
Compute an Eulerian circuit C in G';
Convert the circuit C into a tour T by skipping over previously visited vertices using shortcuts;
return T;

C. Energy-Aware Routing

Let $\mathcal{D}$ be a set of closed disks in the plane whose radii is B. We define α-disk graph as the intersection graph of the disks in $\mathcal{D}$ where every vertex corresponds to a disk in $\mathcal{D}$ and an edge exists between two vertices if the corresponding two disks intersect. In this section, we are tasked with planning a path while following the tour T such that the robot never runs out of energy. As two *consecutive* locations $u, v \in T$ might be further apart than the budget B, the robot needs to travel through one or more charging stations to reach v from u. Let $G_e(V_e, E_e, W_e)$ be an α-disk graph where $\alpha = B$, the node set $V_e := \{u, v\} \cup C$, W_e denotes the weights of the edges such that $w_e(v_e^i, v_e^j)$ is calculated by the Euclidean distance between the node pair $v_e^i, v_e^j \in V_e$, and E_e denote the edge set $\{\{v_e^i, v_e^j\} | w_e(v_e^i, v_e^j) \leq B\}, \forall v_e^i, v_e^j \in V_e$. Next, we call the popular A^* algorithm [16] on every consecutive pair of such nodes in T on G_e and the resulting tour (T_e) never violates the energy constraint. We have used Euclidean distance as our admissible and consistent heuristic function for A^*. We present the pseudo-code in Algorithm 3. A sample visualization by merging all the TSP pair paths is presented in Fig. 3. Node 37 to 38 is an example of where energy-awareness is important – there is no direct edge between these two nodes, and therefore, the robot's path is routed via charging station 17. One should note that when an A^* path between two such consecutive nodes in T are being calculated, only those two nodes from T are present in G_e, and not all the TSP points. This is illustrated when the robot calculates the path between nodes 27 and 28, which is routed via node 1. Since A^* is used to find energy-aware paths between every two successive locations u, v in T, we get optimal length paths between every u, v. The overall solution pseudo-code is presented in Algorithm 4.

Algorithm 3: COMPUTE-ETOUR(T, B, C)

$T_e = \emptyset$;
for all consecutive locations $u, v \in T$ **do**
  Let G_e be a connected graph on $\{u, v\} \cup C$ such that the maximum edge weight is B; $T'_e = A^*(u, v)$ on G_e;
  if T'_e is not NULL **then**
    $T_e = T_e \cup T'_e$;
  else
    return NULL;
  end if
end for
return T_e;

Algorithm 4: The Main Algorithm

Input: S: A set of POIs on a plane.
B: The robot's energy budget.
C: A set of charging stations in the environment.
s: A start point for the energy-aware tour.
Output: T_e: An energy-aware tour;
$\mathcal{L}$ =COVER-POIS(S);
T =COMPUTE-TOUR($\mathcal{L} \cup s$);
T_e = COMPUTE-ETOUR(T, B, C);
return T_e;

[2]In the worst case, the Euclidean length of the tour returned by the Christofides algorithm is 1.5 times that of an optimal tour.

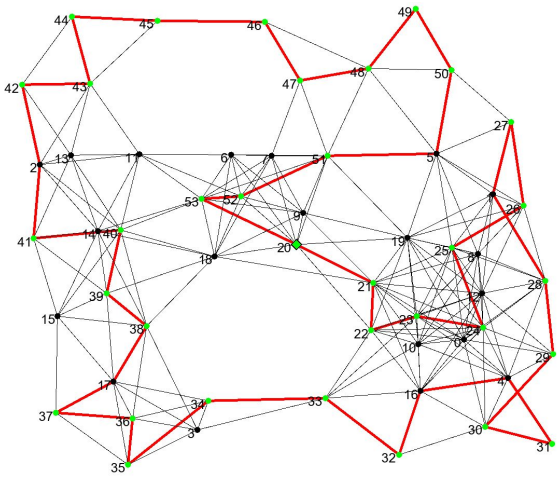


Fig. 3. An illustration of an energy-aware tour T_e is presented. The red path indicates such a tour and the green nodes represent the points in T. The set of charging stations C is represented using the black nodes. The start location s is node 20.

V. Experiments

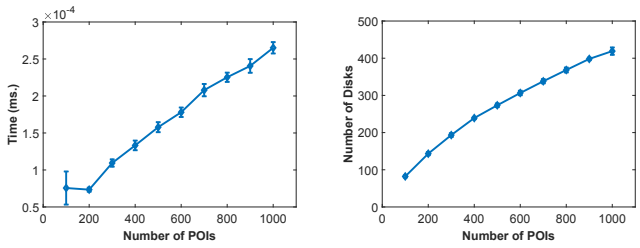
A. Settings

We have implemented the proposed technique using C++ (Algorithm 1) and Java (Algorithms 2 and 3) programming languages. The implementation and testings are performed on a Ubuntu Linux 20.04 LTS desktop equipped with a Ryzen 1600 processor runing at 3.2 GHz. Intel CPU and 8 GB RAM. For an implementation of the Christofides algorithm, we have used Google's OR-Tools. We have used the GraphStream library for graph-related computations and visualizations in Algorithms 2 and 3. The number of POIs n has been varied between $[100, 1000]$ and these POIs have been generated uniformly in a 50×50 unit^2 square environment E, which has the center at $(0, 0)$. The center is also set as the start location s. The energy budget of the robot is varied between $\{8, 10\}\%$ of the maximum distance between any two points in the square environment, i.e., the diagonal of the square. The number of charging stations has been varied between $\{5, 7\}\%$ of the area of the square. Numerical results for 10 valid runs are presented next where the line plots indicate the average values and the error bars represent the standard deviations. For ease of calculations, we have set the visibility radius r to 1. However, as mentioned earlier, our setup can be used for any positive value of r.

B. Results

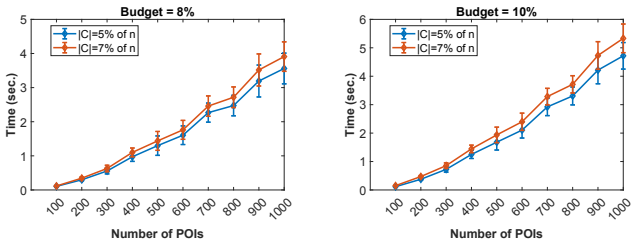
In Fig. 4(a), we show how the runtime of the covering algorithm varies with the increase in POIs. Since the covering algorithm runs in $O(n)$ time on average (as mentioned earlier), we see almost a linear curve in this case. Next, in Fig. 4(b), we show how the number of disks required to cover the POIs vary with the increase of POIs. It is obvious with the increase in the number of POIs, the required number of disks will also increase.

Next, we are interested in investigating the effect of varying the budget amount and the number of charging



(a) (b)

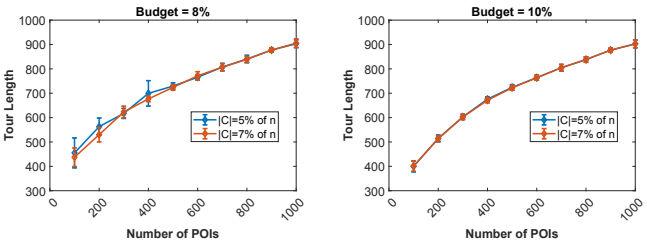
Fig. 4. a) Run time to calculate disk centers using Algorithm 1; b) Number of disk centers found by Algorithm 1 for various POI counts.



(a) (b)

Fig. 5. Run time of Algorithms 2 and 3 with budget a) 8%, and b) 10%.

stations on the run time of the energy-aware tour planning technique. Fig. 5 presents the results for the run time metric. We observe three interesting trends: 1) the run time increases with the number of POIs regardless of the budget and the charging station count, 2) the run time is generally higher with more charging stations in the environment, and 3) with a higher budget, the run time increases. We can explain trend (1) with the fact that as the number of POIs increases, the number of disks increases (Fig. 4(b)) and so does the number of points in the TSP tour. As our technique applies A^* on consecutive TSP locations, the more locations it involves, the higher run time it will incur. The second observation (2) is due to the fact that with more charging stations in the environment, the size of the G_e graph is larger, i.e., more nodes and more edges to consider for every call of A^*. We can observe trend (3) because with a higher budget, more nodes will share edges in G_e, which consequently will make G_e denser. This will result in longer execution times for the A^* algorithm between any two consecutive points on the TSP tour.



(a) (b)

Fig. 6. Lengths of the energy-aware tours calculated by Algorithm 3 with budget a) 8%, and b) 10%.

Next we analyze the tour lengths of the solutions that

our algorithm has found. We observe that the tour costs are usually higher with a lower number of charging stations in the environment. If there are not many charging stations, then the edge weights in the graph G_e are higher as the stations are more sparsely distributed while keeping G_e connected. Therefore, the robot needs to travel longer distances to get recharged. For example, with the number of charging stations $|C|$ set to 5 and 7%, $B = 8\%$, and $n = 200$, the tour lengths are 562.7 and 529.9 units respectively (Fig. 6.(a)). With an increasing budget, however, this difference diminishes. For example, with the same setting and budget increasing to 10%, these values are 514.5 and 513.5 units respectively. Furthermore, we can observe that with more POIs present in the environment, the difference between the tour costs with $|C| = 5$ and 7% also reduces. If there are more POIs in the environment, then there will be more disk centers, and consequently, more points on the TSP tour (see Fig. 4(b) for reference). Therefore, the distances between two consecutive TSP points also reduces due to the uniform POI distribution. Thus, the tour lengths are shorter and they do not rely significantly on the charging stations.

VI. Conclusions

We present a minimalist coverage and energy-aware tour planning solution for an autonomous mobile robot. The robot is assumed to have a fixed sensor radius and its goal is to minimize the number of locations that it needs to visit to provide service to a given set of points. Next, it calculates a tour through this disk centers while starting and ending at point s. However, as the robot might have limited energy, it needs to stop at one or more charging stations in between two successive points of the tour. Our proposed technique is fast and efficient while easily scaling up to 1000 points of interest. In some cases, the graph G_e formed using the charging stations and two disk centers can be dense. Consequently, A* will run slow on G_e. To speed up, in our future work, we will investigate techniques for using linear-sized subgraphs of G_e instead by sacrificing the optimality of the computed paths.

References

[1] E. U. Acar, H. Choset, and J. Y. Lee. Sensor-based coverage with extended range detectors. *IEEE Transactions on Robotics*, 22(1):189–198, 2006.

[2] D. Bhadauria and V. Isler. Data gathering tours for mobile robots. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3868–3873. IEEE, 2009.

[3] S. Bhattacharya, S. Candido, and S. Hutchinson. Motion strategies for surveillance. In *Robotics: Science and Systems*. Citeseer, 2007.

[4] F. Bullo, E. Frazzoli, M. Pavone, K. Savla, and S. L. Smith. Dynamic vehicle routing for robotic systems. *Proceedings of the IEEE*, 99(9):1482–1504, 2011.

[5] N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

[6] A. Dumitrescu, A. Ghosh, and C. D. Tóth. Online unit covering in euclidean space. *Theoretical Computer Science*, 809:218–230, 2020.

[7] A. Dumitrescu and J. S. Mitchell. Approximation algorithms for tsp with neighborhoods in the plane. *Journal of Algorithms*, 48(1):135–159, 2003.

[8] A. Dumitrescu and C. D. Tóth. Constant-factor approximation for tsp with disks. In *A Journey Through Discrete Mathematics*, pages 375–390. Springer, 2017.

[9] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto. Optimal packing and covering in the plane are np-complete. *Information processing letters*, 12(3):133–137, 1981.

[10] R. Friederich, M. Graham, A. Ghosh, B. Hicks, and R. Shevchenko. Experiments with unit disk cover algorithms for covering massive pointsets. *arXiv preprint arXiv:2205.01716*, 2022.

[11] E. Galceran and M. Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276, 2013.

[12] K. Genova and D. P. Williamson. An experimental evaluation of the best-of-many christofides' algorithm for the traveling salesman problem. *Algorithmica*, 78(4):1109–1130, 2017.

[13] A. Ghosh, B. Hicks, and R. Shevchenko. Unit disk cover for massive point sets. In *International Symposium on Experimental Algorithms*, pages 142–157. Springer, 2019.

[14] B. Grocholsky, J. Keller, V. Kumar, and G. Pappas. Cooperative air and ground surveillance. *IEEE Robotics & Automation Magazine*, 13(3):16–25, 2006.

[15] G. Gutin and A. P. Punnen. *The traveling salesman problem and its variations*, volume 12. Springer Science & Business Media, 2006.

[16] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[17] A. R. Karlin, N. Klein, and S. O. Gharan. A (slightly) improved approximation algorithm for metric tsp. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 32–45, 2021.

[18] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[19] D. G. Macharet and M. F. Campos. A survey on routing problems and robotic systems. *Robotica*, 36(12):1781–1803, 2018.

[20] T. Oksanen and A. Visala. Coverage path planning algorithms for agricultural field machines. *Journal of field robotics*, 26(8):651–668, 2009.

[21] M. Schneider, A. Stenger, and D. Goeke. The electric vehicle-routing problem with time windows and recharging stations. *Transportation science*, 48(4):500–520, 2014.

[22] G. Sharma, A. Dutta, and J.-H. Kim. Optimal online coverage path planning with energy constraints. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1189–1197, 2019.

[23] A. Sipahioglu, G. Kirlik, O. Parlaktuna, and A. Yazici. Energy constrained multi-robot sensor-based coverage path planning using capacitated arc routing approach. *Robotics and Autonomous Systems*, 58(5):529–538, 2010.

[24] B. Sotolongo, A. Dutta, S. Sisley, and G. Sharma. Shortest path planning with an energy-constrained robot. In *2021 IEEE International Conference on Systems, Man and Cybernetics, SMC 2021*. IEEE.

[25] F. Suárez-Ruiz, T. S. Lembono, and Q.-C. Pham. Robotsp–a fast solution to the robotic task sequencing problem. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1611–1616. IEEE, 2018.

[26] K. Sundar and S. Rathinam. Algorithms for routing an unmanned aerial vehicle in the presence of refueling depots. *IEEE Transactions on Automation Science and Engineering*, 11(1):287–294, 2013.

[27] O. Tekdas, V. Isler, J. H. Lim, and A. Terzis. Using mobile robots to harvest data from sensor fields. *IEEE Wireless Communications*, 16(1):22–28, 2009.

[28] R. van Bevern and V. A. Slugina. A historical note on the 3/2-approximation algorithm for the metric traveling salesman problem. *Historia Mathematica*, 53:118–127, 2020.

[29] J. I. Vasquez-Gomez, J.-C. Herrera-Lozada, and M. Olguin-Carbajal. Coverage path planning for surveying disjoint areas. In *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 899–904. IEEE, 2018.

[30] M. Wei and V. Isler. Coverage path planning under the energy constraint. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 368–373. IEEE, 2018.

[31] J. Xie, L. R. G. Carrillo, and L. Jin. An integrated traveling salesman and coverage path planning problem for unmanned aircraft systems. *IEEE control systems letters*, 3(1):67–72, 2018.