

# Optimal Sequential Stochastic Deployment of Multiple Passenger Robots

Chris (Yu Hsuan) Lee, Graeme Best, Geoffrey A. Hollinger

**Abstract**—We present a new algorithm for deploying passenger robots in marsupial robot systems. A marsupial robot system consists of a carrier robot (e.g., a ground vehicle), which is highly capable and has a long mission duration, and at least one passenger robot (e.g., a short-duration aerial vehicle) transported by the carrier. We optimize the performance of passenger robot deployment by proposing an algorithm that reasons over uncertainty by exploiting information about the prior probability distribution of features of interest in the environment. Our algorithm is formulated as a solution to a sequential stochastic assignment problem (SSAP). The key feature of the algorithm is a recurrence relationship that defines a set of observation thresholds that are used to decide when to deploy passenger robots. Our algorithm computes the optimal policy in  $O(NR)$  time, where  $N$  is the number of deployment decision points and  $R$  is the number of passenger robots to be deployed. We conducted drone deployment exploration experiments on real-world data from the DARPA Subterranean challenge to test the SSAP algorithm. Our results show that our deployment algorithm outperforms other competing algorithms, such as the classic secretary approach and baseline partitioning methods, and is comparable to an offline oracle algorithm.

## I. INTRODUCTION

Exploration of increasingly complex environments demands more flexible robotic capabilities. Developments in heterogeneous multi-robot systems have yielded carrier-passenger robot systems called “marsupial robots”, in which highly-capable carrier robots transport and deploy one or more low-capability passenger robots. These marsupial robots can tailor their complementary capabilities to the challenges of exploring complex environments [2]. During exploration, these environments can contain multiple features of interest that the carrier robot may want to observe but are prohibitive or difficult to reach [3], [4]. In marine environments, a large ship may carry and deploy multiple heterogeneous robots to increase the rate of information gathering of features such as seafloor mines, adversarial vessels, and biological hotspots [5], [6], [7]. In the case of urban environments, a team of ground and aerial robots may seek to explore features like ledges, vertical shafts, stairways, or other urban features that ground robots cannot easily reach [8]. During these missions, the carrier robot would ideally deploy the passenger robot at a location that can maximize exploration coverage or information gain and make

\*Approved for public release; distribution is unlimited. This work was in part sponsored by DARPA under agreement #HR00111820044 and Office of Naval Research Grant N00014-17-1-2581. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

\*A preliminary version of this work appeared as a workshop paper in [1].

\*The authors are with the Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University, Corvallis OR, USA. {leeyuh, bestg, geoff.hollinger}@oregonstate.edu

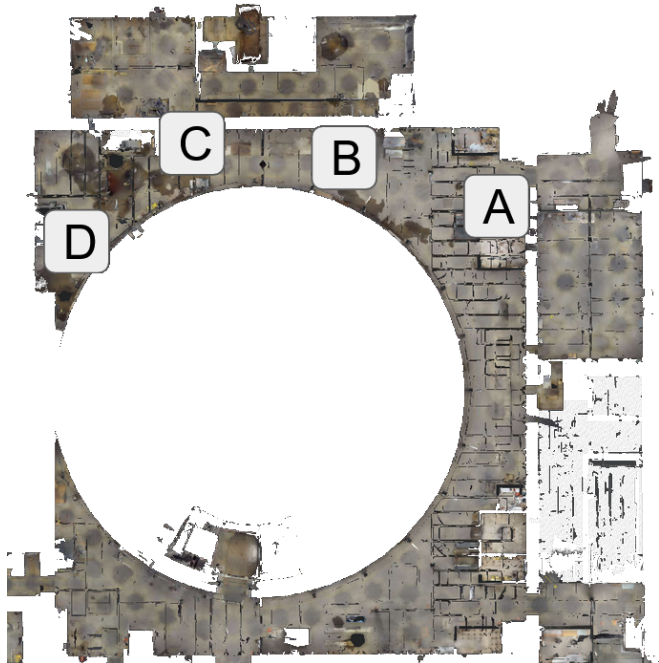


Fig. 1. RGB point cloud map [9] of an urban environment from the DARPA Subterranean Challenge [10] with example deployment locations. Each decision point is revealed sequentially to the robot. Point A has high exploration potential since it leads to another room, but not as much as point C, which leads to multiple rooms and a stairwell only accessible to aerial passenger robots. However, points B and D are poor deployment locations since they lack nearby features to be explored by the passenger robots.

use of the passenger robot’s complementary capabilities (e.g., flying in 3D).

In order to optimize exploration efficiency, a marsupial robot system must decide when and where to deploy its passenger robots. In Fig. 1, an example of an urban environment is illustrated with sequential decision locations having different exploration potential. If a carrier robot deploys too early, it risks missing out on the potentially more valuable later decision points. Each of these sequential decision locations may have different reward values that are only revealed when they are observed along the robot’s path. Due to the unknown nature of an unexplored environment, the carrier robot is required to make online deployment decisions while reasoning over the possible discovered deployment value at each potential deployment location. Furthermore, the value and number of these features are not known in advance. Multi-robot systems require coordination beyond naive partitioning of deployment between the passenger robots to ensure that the maximum expected number of features are captured [11]. Prior works have explored marsupial robot coordination and planning [4], [12] but, to our knowledge, only deploy in a

naive manner [13]. To address these challenges, we develop an online passenger deployment strategy that reasons over the predicted future reward of deploying at each decision point.

We formulate the passenger robot deployment problem as a sequential stochastic assignment problem (SSAP) [14] where a set of passenger robots are assigned to deployment locations. The deployment algorithm exploits prior probabilistic information of the distribution of features throughout the environment. The key component of the algorithm is a recurrence relationship that defines a set of observation thresholds. These thresholds are used to decide when to deploy passenger robots by comparing to the current observation of deployment reward. The algorithm is guaranteed to find the optimal online deployment policy for problems with a known prior distribution with independent observations. Also, the algorithm is efficient in that it has  $O(NR)$  runtime, where  $N$  is the number of decision points and  $R$  is the number of passenger robots. This computational complexity improves upon [14], which had  $O(N^2)$  complexity.

We conducted drone deployment exploration simulated experiments where a ground robot seeks to deploy drones in exploration-valuable deployment locations. The ground robot observes hard-to-reach frontier cells that are valuable for exploration by the aerial drone in each deployment location and must make a decision to deploy or not deploy the aerial drone. Data from the DARPA Subterranean (SubT) Challenge Urban circuit [15] were used for the experiments. We show empirically that our algorithm is competitive with an offline oracle solution that has full access to the rewards in advance. Our algorithm also greatly outperforms comparable deployment algorithms, including a partitioned solution to the classic secretary problem [16], and naive baseline methods. In the experiment, the algorithm is shown to be capable of choosing to deploy aerial drones at exploration-valuable deployment locations. On average, the SSAP algorithm performed within 96% of the oracle in experiments with simulated data and performed within 90% of the oracle on experiments with the real-world data.

Our contributions in this paper are:

- 1) A multi-robot passenger deployment problem formulated as a sequential stochastic assignment problem
- 2) An informed passenger deployment strategy for this problem, which builds on [14], that maximizes the expected sum of the deployment rewards.

## II. RELATED WORK

Traditional heterogeneous robot systems have loosely-coupled motion constraints that are not physically constricted and have been coordinated with distributed planners [17]. In marsupial robots, tightly-coupled constraints are physically imposed on the deployments of the passenger robots by the carrier robots. The increased complexity requires new planners that are able to handle tasks like coordination, deployment, retrieval, and manipulation. Some marsupial robot planners are able to organize and coordinate between multiple marsupial robotic systems to handle exploratory

navigation actions [4], [12] but do not explicitly decide on optimal deployment locations for passenger robots. Other planners require known potential routes and deployment locations to decide optimal passenger robot deployment locations [18]. Another marsupial robot planner variant simplifies the deployment decision by deploying single passenger robots at fixed times such as the beginning of the mission [19]. Finally, prior domain knowledge, such as marine flow fields [20], has been used to inform multiple passenger robot planners to maximize information gain. However, these types of historical information field data may not be readily available for other domains such as urban environments. We present an algorithm that is capable of making multiple explicit passenger robot deployment decisions without a need to know potential routes.

Our deployment problem is closely related to optimal stopping theory [16], which considers problems that require deciding online the right time to carry out a particular action. A key example of an optimal stopping problem is the Classic Secretary problem, but it does not leverage prior information regarding future rewards [16]. However, the Cayley–Moser problem [21] reasons over predicted future rewards by considering the prior probability distribution. This has been applied to robotics problems by Lindhé and Johansson [22], who consider the problem of when to communicate by utilizing a multi-path fading communication model to make predictions about future stopping decisions. Our method is a generalization of the Cayley–Moser algorithm, adapted for the context of the deployment problem. Additionally, we formulate a policy that considers all future deployments, rather than just the single next decision point.

Optimal stopping variants have been generalized to multiple decision points. Das et al. [5] apply multi-choice optimal stopping theory for AUVs to collect multiple plankton samples that minimize the cumulative regret of the samples. They explore two applicable algorithms, the multi-choice hiring algorithm [23] and the submodular secretary algorithm [24]. Both of these algorithms seek to select the best observations and do not consider information such as the distribution of the observations. The sequential stochastic assignment problem (SSAP) [14] generalizes the Cayley–Moser algorithm and finds an optimal policy to maximize the expected sum of rewards from multiple assignments of agents to tasks. SSAP has been applied to other fields [25] but, to our knowledge, has not been utilized in robotics. A closely related body of work focuses on multi-robot task assignment [26]; however, these problems generally assume that task values are known a priori and are therefore not directly applicable to online problems, such as ours.

## III. PROBLEM FORMULATION

We consider a marsupial robot system that must make online decisions regarding when to deploy its passenger robots. At each possible deployment location, the robot must consider if the reward gained from deployment is expected to be more favorable than continuing onwards and deploying

at a later location. We formulate the multi-robot deployment decision as a sequential stochastic assignment problem (SSAP). Multiple deployment decisions are made based on sequentially revealed random variables. We formalize this problem as follows.

We consider an environment that contains a set of point features, which may represent points of interest that are ideal for additional exploration by passenger robots, but are ill-suited for the carrier robot. As the carrier robot moves through the environment, it must make decisions to deploy or not deploy a passenger robot. There are assumed to be a total of  $N$  decision points, where  $N$  is known to the robot. The carrier robot houses  $R$  passenger robots of equal capability. At each decision point, the carrier robot may choose to deploy one passenger robot. Along the sequence of decision points, the independent observations of the number of features in the sensing area are denoted  $(X_1, \dots, X_N)$ . Furthermore, the algorithm relies on knowing a prior distribution of the random variable  $X$ , denoted as  $f(x)$ . Later in Sec. V, we present an example where the features of interest correspond to the number of exploration frontier cells observed by the ground robot with a prior distribution  $f(x)$  defined as a Poisson distribution.

At stage  $j \in \{1, \dots, N\}$ , the carrier robot reaches a decision point, and the outcomes of all random variables  $(X_1, \dots, X_j)$ , denoted  $(x_1, \dots, x_j)$ , are known to the robot. Along the path, the robot must make an irreversible decision at each deploy location to deploy or continue on to deploy later. If the carrier robot decides to deploy, it assigns one passenger robot to the deployment location and returns a reward of  $x_j$ . If the carrier robot decides to continue, no reward is claimed at this stage. This process continues for the  $N$  stages. All passenger robots must be deployed by stage  $N$ , with the constraint of one deployment per stage.

We define the set of stage indices where passenger robots were deployed as  $(d_1, \dots, d_R)$ . The goal of the carrier robot is to maximize the expected sum of the reward returned from the deployment locations; i.e., find the optimal deployment sequence:

$$(d_1, \dots, d_R)^* = \underset{(d_1, \dots, d_R)}{\operatorname{argmax}} \mathbb{E} \left[ \sum_{r=1}^R x_{d_r} \right]. \quad (1)$$

Here,  $x_{d_r}$  denotes the reward from passenger robot  $r$  and the expectation is taken over the distribution  $f(x)$ .

#### IV. ONLINE PASSENGER DEPLOYMENT ALGORITHM

We present the online passenger deployment algorithm, which finds the optimal deployment policy in linear time that scales with the number of passenger robots and deployment locations. The algorithm is computed via dynamic programming, using a technique that stems from sequential stochastic assignment [14]. We consider the general case of  $R$  passenger robots to deploy. Finally, we provide an analysis of the optimality and runtime complexity of the algorithm.

##### A. Multi-Robot Deployment

Our deployment algorithm precomputes a set of thresholds, which are a function of the distribution of the observations and the number of remaining stages,  $n$ . The current observed value  $x_j$  at stage  $j$  is compared to the threshold values and informs the carrier robot whether or not to deploy now. At each stage  $j$ , we define a sequence of values  $(p_1, \dots, p_n)$ , such that  $p_i = 1$  for  $n - r < i \leq n$ , representing the remaining  $r$  passenger robots, and  $p_i = 0$  for  $1 \leq i \leq n - r$ . This can be thought of as assigning a  $p_i$  at each stage to each  $x_j$ .

Specifically, the optimal policy for the passenger robot assignment is to assign  $p_i$  to the observed deployment value  $x_j$ , if  $x_j$  falls into an  $i$ th non-overlapping interval comprising the real line [14]. These non-overlapping intervals are separated by a set of thresholds, denoted as  $a_{i,n}$ . Each threshold  $a_{i,n}$  is computed recursively and depends only on the prior distribution  $f(x)$ , as well as the number of remaining stages  $n$ . For stage  $j$ , where there are  $n = N - j + 1$  stages remaining, there are a set of  $n + 1$  thresholds, such that

$$-\infty = a_{0,n} \leq a_{1,n} \leq a_{2,n} \leq \dots \leq a_{n,n} = \infty. \quad (2)$$

At stage  $j$ , the optimal choice is to assign  $p_i$  if the realization  $x_j$  of the random variable  $X_j$  is contained in  $(a_{i-1,n}, a_{i,n}]$ . For example, given  $n = 4$  remaining stages and  $r = 2$  robots to deploy,  $p_3, p_4 = 1$  and  $p_1, p_2 = 0$ . If the observed value of  $x_j$  is  $a_{2,4} < x_j \leq a_{3,4}$ , then the decision is to deploy and returns a reward of  $p_3 x_j = x_j$ . At the next stage with  $n = 3$  and  $r = 1$ , we only have  $p_3 = 1$  and  $p_1, p_2 = 0$ . If the next observed value  $x_j < a_{2,3}$ , then there is no deploy action and returns a reward of  $p_2 \cdot x_j = 0$ .

An assignment of  $p_i = 1$  to  $x_j$  results in a deployment of the passenger robot whereas an assignment of  $p_i = 0$  to  $x_j$  leads to a non-deployment action. There is no need to differentiate between interval containment below the deploy threshold  $a_{i-R,n}$  since all of the  $n - R$  assignments of  $p_i = 0$  result in the same non-deployment behavior. Thus, only  $R$  thresholds, instead of  $n + 1$  as shown in (2), need to be calculated for each stage  $n$  beyond  $n = R$ , and this reduces the complexity from quadratic [14] to linear in  $N$ , as discussed later in Sec. IV-B.

The threshold  $a_{i,n+1}$  is defined as the expected value, if there are  $n$  stages remaining, of the quantity to which the  $i$ th smallest  $p$  is assigned [14]. This formulates the recurrence relationship

$$\begin{aligned} a_{i,n+1} &= \Pr(x_n < a_{i-1,n}) a_{i-1,n} \\ &\quad + \Pr(a_{i-1,n} < x_n < a_{i,n}) \\ &\quad \times \mathbb{E}(x_n | a_{i-1,n} < x_n < a_{i,n}) \\ &\quad + \Pr(x_n > a_{i,n}) a_{i,n} \end{aligned} \quad (3)$$

$$\begin{aligned} &= a_{i-1,n} \int_{-\infty}^{a_{i-1,n}} f(x) dx + \int_{a_{i-1,n}}^{a_{i,n}} x f(x) dx \\ &\quad + a_{i,n} \int_{a_{i,n}}^{\infty} f(x) dx, \end{aligned} \quad (4)$$

4	-	-	-	$\infty$
3	-	-	$\infty$	$a_{3,4}$
2	-	$\infty$	$a_{2,3}$	$a_{2,4}$
1	$\infty$	$a_{1,2}$	$a_{1,3}$	-
0	$-\infty$	$-\infty$	$-\infty$	$-\infty$
$i \backslash n$	1	2	3	4

Fig. 2. The method for the recurrence threshold calculation and terms relationship in (3), for  $R = 2$  and  $N = 4$ .  $a_{i,n}$  is the threshold for the  $i$ th non-overlapping interval when there are  $n$  stages left to go. The term  $a_{2,3}$  is calculated from the  $a_{i-1,n}$  term, which is  $a_{1,2}$ , and the  $a_{i,n}$  term, which is  $a_{2,2} = \infty$ .  $a_{1,4}$  does not need to be calculated since  $R = 2$ .

**Algorithm 1** SSAP optimal policy threshold calculation for passenger robot deployment

**Input:** Number of stages:  $N$ , Number of robots:  $R$ ,

Prior distribution:  $f(x)$

**Output:** SSAP Thresholds:  $a_{i,j}$

```

1: for  $j = 0, 1, 2, \dots, N$  do
2:    $a_{j,j} = \infty$ 
3:   for  $i = j - R, \dots, j - 1$  do
4:      $a_{i,j} \leftarrow \text{Eqn. (4)}$ 
```

where  $-\infty \cdot 0 = 0$  and  $\infty \cdot 0 = 0$ . In both (3) and (4), the second term is for the case where  $x_n$  lies within the  $i$ th interval, and therefore  $p_i$  receives the value of  $x_n$ . The first term is for the case where  $x_n$  is below the  $i$ th interval, meaning that  $x_n$  is assigned to a  $p_k < p_i$ , and thus  $p_i$  is assigned in a later stage with an expected value of  $a_{i-1,n}$ . Similarly, the third term is for the case where  $x_n$  is above the interval and  $p_i$  has an expected future assignment of  $a_{i,n}$ .

The recurrence process is illustrated in Fig. 2, where the cells in the table can be computed from left to right by reusing the previous values, as indicated by the arrows. Furthermore, an outline of the algorithm is provided in Alg. 1. The outer loop iterates through the  $N$  stages while the inner loop iterates through only  $R$  thresholds. The algorithm's recurrence process has a computational complexity of  $O(NR)$ . In the  $R = 1$  single passenger robot deployment case, the threshold calculations are identical to the thresholds used in the Cayley–Moser optimal stopping problem [21].

### B. Analysis

The dynamic programming proceeds by iteratively solving optimal subproblems for  $a_{i,n+1}$  using the recurrence relationship in (4), and thus computes the optimal set of thresholds  $a_{i,n}, \forall i, n$ . The full proof of the results that these subproblems are optimal, and that these thresholds lead to an optimal online assignment policy, is presented in [14]. The proof proceeds by induction, and relies on Hardy's Theorem [27], which states that the optimal assignment between two sets with a sum-product objective is to pair the smallest values in each set, then the next smallest, and so on until the largest values are paired.

As illustrated in Fig. 2 and Alg. 1, there are  $O(NR)$  sub-problems to be computed, where  $N$  is the number of stages and  $R$  is the number of passenger robots to deploy. The integral (4) is computed once per sub-problem, thus the computation time is  $O(NRF)$ , where  $F$  is the time to compute (4).

## V. EXPERIMENTS AND RESULTS

In order to evaluate our proposed SSAP passenger robot deployment algorithm, we performed drone deployment exploration experiments using simulated data and real-world data from the DARPA SubT challenge, against various other deployment strategies.

### A. Comparison Methods

Different deployment strategies were employed, each aiming to maximize the number of features captured during drone deployment, to test the efficacy of our SSAP deployment algorithm. These deployment strategies were adapted from optimal stopping algorithm variants to explicitly handle multiple robot deployments. As such, for these comparison methods, the carrier robot path are naively divided into  $R$  equal partitions in the case of multi-robot deployment and treat each partition independently from each other. The deployment algorithms are described below:

- *Sequential Assignment (SSAP)*: Performs our method as described in Sec. IV-A and considers the entire path as a singular deployment, without partitions. The prior distributions are described in Sec. V-B.1 and Sec. V-C.1.
- *Oracle*: Selects the top  $R$  deployment locations across all the partitions, with perfect knowledge of the world in advance.
- *Partition Oracle*: Selects the best deployment location within each partition, with perfect knowledge of the world in advance.
- *Partition Cayley–Moser*: Performs as described in Sec. IV-A when  $R = 1$ , in each partition.
- *Partition Classic Secretary*: An optimal stopping variant that only observes for the first  $N/eR$ , where  $e$  is Euler's number, of decision points and then selects the next value that is higher than the max value observed in the observation phase [16]. The algorithm runs within each partition.
- *Random*: Selects a decision point randomly within each partition.

The experiments were conducted on a standard desktop computer, with an i5-4690K CPU and 16 GB of RAM.

### B. Capturing Poisson-Distributed Features

1) *Experimental Setup*: A carrier robot travels through a 2D simulated world containing features of interest distributed as a stationary Poisson point process. At decision point  $j$ , the carrier robot detects features within a circular sensing area centered at the current location. The number of features  $x_j$  detected within a sensing area are modeled as a Poisson distribution with probability mass function

$$f(x) = \frac{\lambda^x e^{-\lambda}}{x!}, \text{ for } x \in \{0, 1, \dots\}. \quad (5)$$

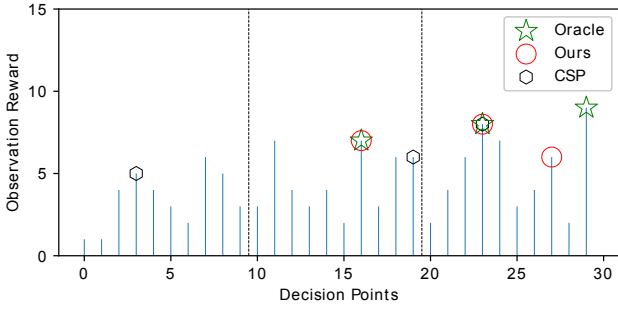


Fig. 3. Illustrative example of the deployments for three passenger robots to be deployed over 30 decision points. Our method (reward: 21) performs similar to the *Oracle* (reward: 24) that has full knowledge of the observation sequence in advance. The *Classic Secretary* (reward: 19) algorithm performed worse, in part due to being constrained by having one deployment in each of the three partitions (dotted lines).

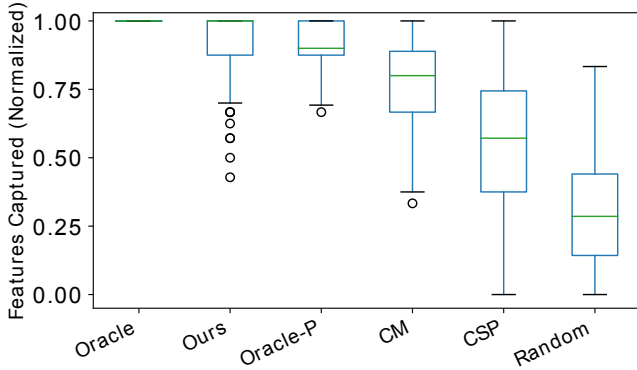


Fig. 4. Algorithm comparisons with three passenger robots over 150 trials. *Partition Oracle* is denoted as Oracle-P. CM is *Cayley-Moser* and CSP is *Classic Secretary*.

Here, the Poisson distribution  $f(x)$  in (5) is an example of a distribution used in the threshold calculations in (4). The rate of this Poisson distribution,  $\lambda$ , is the expected number of features of interest in the robot sensing area. The *SSAP* thresholds are calculated by substituting (5) into (4), yielding

$$a_{i,n+1} = a_{i-1,n} \sum_{x=0}^{\lfloor a_{i-1,n} \rfloor} \frac{\lambda^x e^{-\lambda}}{x!} + \sum_{x=\lfloor a_{i-1,n} \rfloor}^{\lfloor a_{i,n} \rfloor} x \frac{\lambda^x e^{-\lambda}}{x!} + a_{i,n} \left[ 1 - \sum_{x=0}^{\lfloor a_{i,n} \rfloor} \frac{\lambda^x e^{-\lambda}}{x!} \right]. \quad (6)$$

2) *Results*: An example of a trial of the algorithm selection process is illustrated in Fig. 3. The *Oracle* algorithm optimally selects the top  $R = 3$  rewards. *SSAP* decided to deploy and select the reward in an optimal fashion in the first two decision cases. In comparison, the *Classic Secretary* algorithm does not reason over possible future observations and locally selects a decision point in each partition.

The results of *SSAP* through repeated trials are very encouraging. Trials with three passenger robots and  $n = 60$  stages are shown in Fig. 4. During each trial, the algorithms' results are compared to the maximum possible number of features captured by the *Oracle*. On average, *SSAP* performed within 96% of the *Oracle*. Also, *Partition Oracle*

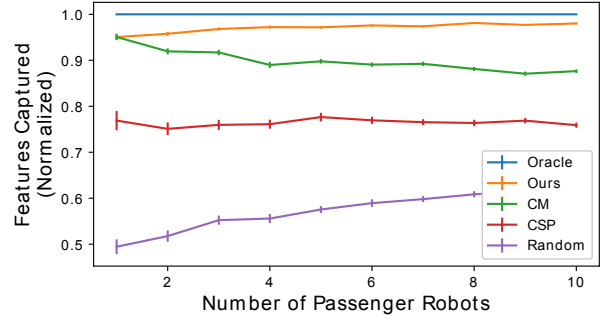


Fig. 5. Algorithm comparisons with average utility, as a percentage of the *Oracle*, over  $R$  passenger robots and 150 trials. SEM error bars are displayed.

is not perfect like the *Oracle* but performs similarly when compared to *SSAP*.

The partitioned *Cayley-Moser* algorithm has a lower performance than *SSAP* since the calculated thresholds are only local to each partition and partitioned algorithms are constrained to one deployment location per partition. *SSAP* is able to account for the expected future values over the total number of stages, whereas the partitioned *Cayley-Moser* locally calculated the thresholds only for the number of stages in a partition.

Lastly, we show the scalability of *SSAP* to  $R$  passenger robots in Fig. 5. The offline computation of the thresholds is linear in  $N$  stages and  $R$  number of robots. In practice, the computation of the thresholds takes on the order of seconds. As  $R$  increases, the performance of *SSAP* clearly outperforms other algorithms. *SSAP* is mathematically equivalent to *Cayley Moser* in the case where  $R = 1$  and yields identical results, as discussed in Sec. IV-A.

### C. Aerial Robot Deployment for Exploration

1) *Experimental Setup*: Recorded LiDAR data from the Urban Circuit of the DARPA Subterranean Challenge [10] provided a real-world scenario to test the efficacy of our passenger deployment algorithm. Multiple drone-carrying ground robots autonomously explored the Satsop Nuclear Power Plant in Elma, Washington. For many teams in the competition, a drone deployment was manually triggered by the operator. Identifying an ideal deployment location required constant attention from the operator, already burdened with other tasks. Furthermore, it would be advantageous to utilize prior knowledge from the first robot entering the environment or information from a similar environment.

An occupancy grid of the world was generated from the collected LiDAR data using OpenVDB [28]. Using these data, a deployment decision location was set every 2.5 meters along the path that the robot travelled throughout the environment. We defined the value of a deployment location based on the total number of frontier cells within a 10 m radius of the ground robot's location. Frontier cells in the occupancy grid are defined to be free cells neighboring at least one unknown cell [29]. Furthermore, we filter out frontier cells that are considered accessible to the ground robot. Only frontier cells that are above 1 m and below



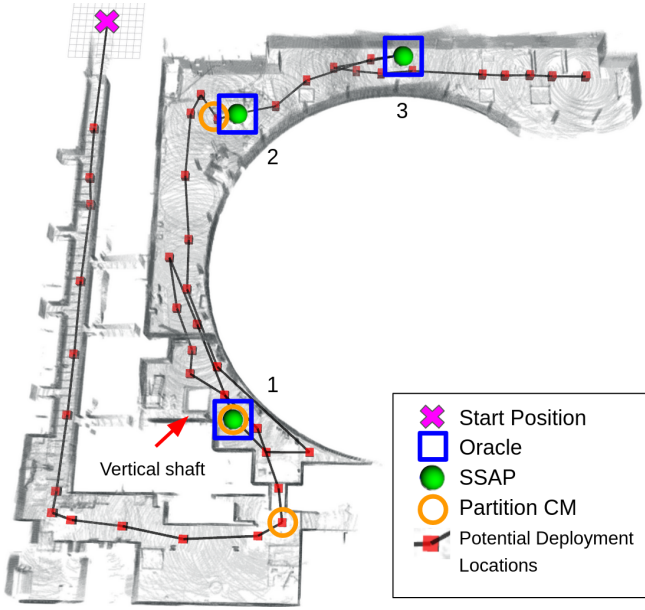


Fig. 6. Deployment locations along a 105 m path in the Alpha environment from the DARPA SubT Urban Circuit. The SSAP algorithm decides to deploy in the optimal locations. Deploy location 1 is adjacent to a vertical shaft, reachable only by an aerial drone. Location 2 has a higher ceiling and is a larger space than the first CM location. Lastly, location 3 leads to the rest of the environment and provides high exploration value.

0.1 m of the ground robot’s position are included. These filtered frontier cells may represent openings such as shafts and ledges that are inaccessible to the ground robots but are ideal for aerial drones.

Different distributions, representing different prior knowledge, for  $f(x)$  in (4) were defined in the calculation of the SSAP thresholds. In addition to the Poisson distribution, we compared two other distributions utilized by SSAP: (1) the histogram distribution and (2) Conway-Maxwell-Poisson (CMP) distribution. First, a distribution was generated from a histogram of the deployment values encountered during the robot’s run and can be considered the ground truth distribution for that specific robot’s path. Secondly, the CMP distribution generalizes the Poisson distribution and can better handle over/under-dispersion [30], with the pmf

$$f(x) = \frac{\lambda^x}{(x!)^\nu} \frac{1}{Z(\lambda, \nu)}, \text{ for } x \in \{0, 1, \dots\} \quad (7)$$

where  $Z(\lambda, \nu)$  is a normalization constant. The parameters  $\nu$  and  $\lambda$  were estimated using a Mean-Squared-Error fit to the histogram data. Similar to the Poisson distribution in Sec. V-B.1, the CMP distribution in (7) can be substituted into (4) to generate the SSAP thresholds.

2) *Results*: The SSAP algorithm performed strongly in the real-world experiments and outperformed the competing deployment algorithms. The CMP distribution was able to better model the higher variability of real-world deployment values than the Poisson distribution. As seen in Fig. 7, SSAP using the CMP distribution performs within 87% of the oracle. While slightly less than the performance of the ground truth histogram distribution (91% of oracle), the SSAP-CMP distribution outperforms the SSAP-Poisson distribution by

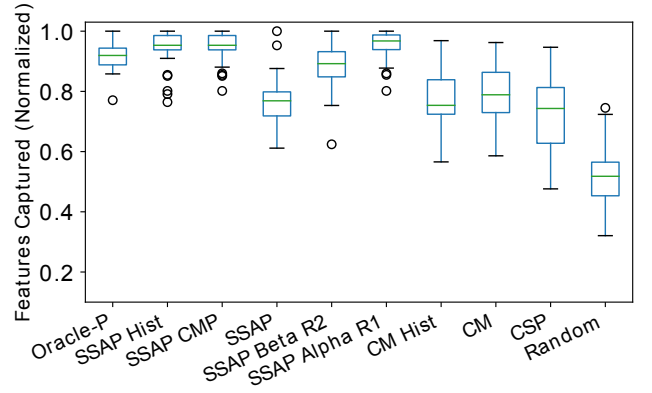


Fig. 7. Algorithm comparison results from real data of a robot (R2) exploring Alpha environment. The SSAP Beta R2 algorithm utilizes histogram data as the prior distribution from the same robot in another environment (Beta). The SSAP Alpha R1 algorithm uses the histogram data from another robot (R1) in the same environment.

18%. For the remaining experiment variations, SSAP-CMP is comparable to the SSAP-Hist algorithm and outperforms SSAP-Poisson by at least 10% and in one case, up to 28%.

We explored two cases of inter-robot information transfer to take advantage of prior domain knowledge: one where the histogram data from one robot is used in another robot in the same environment and another where the data from another environment are used. The results of the same environment, inter-robot histogram transfer (*Alpha R1*) can be seen in Fig. 7. The SSAP algorithm using the histogram distribution from another robot performs comparably to the high-performing CMP and ground truth histogram distribution, above 90% of the oracle on average. In the other case, the histogram distribution from a different environment performed on average around 80% of oracle. Despite a lower performance than inter-robot histogram transfer, the inter-environment histogram transfer (*Beta R2*) can still provide better results than SSAP-Poisson or the other baseline methods, *partition Cayley-Moser* and *partition CSP*.

## VI. FUTURE WORK

We have provided a formulation of passenger robot deployment for marsupial robots and demonstrated the feasibility of the sequential assignment optimal policy to passenger robot deployments. Our results demonstrated that our algorithm enables a significantly improved selection of aerial robot deployment locations for exploration tasks, in comparison to a variety of alternative methods. In the future, it would be interesting to improve current distribution estimation [31], study cases where the prior feature distribution [32] or number of stages are not known, and consider a dependent relationship between consecutive observations. Lastly, we aim to extend the deployment problem formulation to multiple carrier robots as well.

## ACKNOWLEDGMENT

We would like to thank Rohit Garg for help with the frontier extraction implementation and the SubT Team Explorer [15] for inspiration and providing real-robot data.

## REFERENCES

- [1] C. Y. H. Lee, G. Best, and G. A. Hollinger, "Optimal deployment of multiple passenger robots using sequential stochastic assignment," in *Proc. RSS 2020 Workshop on Heterogeneous Multi-Robot Task Allocation and Coordination*, 2020.
- [2] R. R. Murphy, M. Ausmus, M. Bugajska, T. Ellis, T. Johnson, N. Kelley, J. Kiefer, and L. Pollock, "Marsupial-like mobile robot societies," *Proc. Int. Conf. on Autonomous Agents*, pp. 364–365, 1999.
- [3] P. Tokekar, J. Vander Hook, D. Mulla, and V. Isler, "Sensor planning for a symbiotic UAV and UGV system for precision agriculture," *IEEE Trans. on Robotics*, vol. 32, no. 6, pp. 1498–1511, 2016.
- [4] J. Moore, K. C. Wolfe, M. S. Johannes, K. D. Katyal, M. P. Para, R. J. Murphy, J. Hatch, C. J. Taylor, R. J. Bamberger, and E. Tunstel, "Nested marsupial robotic system for search and sampling in increasingly constrained environments," in *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, 2016, pp. 002 279–002 286.
- [5] J. Das, F. Py, J. B. Harvey, J. P. Ryan, A. Gellene, R. Graham, D. A. Caron, K. Rajan, and G. S. Sukhatme, "Data-driven robotic sampling for marine ecosystem monitoring," *Int. J. Robotics Research*, vol. 34, no. 12, pp. 1435–1452, 2015.
- [6] F. Marques, A. Lourenço, R. Mendonça, E. Pinto, P. Rodrigues, P. Santana, and J. Barata, "A critical survey on marsupial robotic teams for environmental monitoring of water bodies," in *Proc. IEEE OCEANS*, 2015.
- [7] M. Kalaitzakis, B. Cain, N. Vitzilaios, I. Rekleitis, and J. Moulton, "A marsupial robotic system for surveying and inspection of freshwater ecosystems," *J. Field Robotics*, 2020, DOI: 10.1002/rob.21957.
- [8] T. Rouček, M. Pecka, P. Čížek, T. Petříček, J. Bayer, V. Šalanský, D. Heřt, M. Petrlík, T. Báča, V. Spurný *et al.*, "DARPA subterranean challenge: Multi-robotic exploration of underground environments," in *Proc. Int. Conf. on Modelling and Simulation for Autonomous Systems*. Springer, 2019, pp. 274–290.
- [9] DARPA, "Explore alpha course in 3D," Accessed on 03.12.2020. [Online]. Available: <https://my.matterport.com/show/?m=dP4NytELbJB>
- [10] B. Allen and T. Chung, "Unearthing the subterranean environment," 2020, Accessed on 03.22.2020. [Online]. Available: <https://www.subtchallenge.com/>
- [11] N. Karapetyan, K. Benson, C. McKinney, P. Taslakian, and I. Rekleitis, "Efficient multi-robot coverage of a known environment," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2017, pp. 1846–1852.
- [12] K. M. Wurm, C. Dornhege, B. Nebel, W. Burgard, and C. Stachniss, "Coordinating heterogeneous teams of robots using temporal symbolic planning," *Autonomous Robots*, vol. 34, no. 4, pp. 277–294, 2013.
- [13] M. S. Couceiro, D. Portugal, R. P. Rocha, and N. M. Ferreira, "Marsupial teams of robots: Deployment of miniature robots for swarm exploration under communication constraints," *Robotica*, vol. 32, no. 7, pp. 1017–1038, 2014.
- [14] C. Derman, G. J. Lieberman, and S. M. Ross, "A sequential stochastic assignment problem," *Management Science*, vol. 18, no. 7, pp. 349–355, 1972.
- [15] CMU/OSU, "SubT Team Explorer Website CMU/OSU," Available at [https://www.subt-explorer.com/\(2020/08/12\)](https://www.subt-explorer.com/(2020/08/12)), Accessed on 03.25.2020.
- [16] T. S. Ferguson, *Optimal Stopping and Applications*. UCLA, 2008, ch. Finite Horizon Problems, p. 2.1–2.15.
- [17] G. Best, W. Martens, and R. Fitch, "Path planning with spatiotemporal optimal stopping for stochastic mission monitoring," *IEEE Trans. on Robotics*, vol. 33, no. 3, pp. 629–646, 2017.
- [18] N. Mathew, S. L. Smith, and S. L. Waslander, "Planning paths for package delivery in heterogeneous multirobot teams," *IEEE Trans. on Automation Science and Engineering*, vol. 12, no. 4, pp. 1298–1308, 2015.
- [19] J. C. Las Fargeas, P. T. Kabamba, and A. R. Girard, "Path planning for information acquisition and evasion using marsupial vehicles," in *Proc. IEEE American Control Conf. (ACC)*, 2015, pp. 3734–3739.
- [20] J. Hansen, S. Manjanna, A. Q. Li, I. Rekleitis, and G. Dudek, "Autonomous marine sampling enhanced by strategically deployed drifters in marine flow fields," in *Proc. MTS/IEEE OCEANS*, 2018.
- [21] L. Moser, "On a problem of Cayley," *Scripta Math*, vol. 22, pp. 289–292, 1956.
- [22] M. Lindhé and K. H. Johansson, "Exploiting multipath fading with a mobile robot," *Int. J. Robotics Research*, vol. 32, no. 12, pp. 1363–1380, 2013.
- [23] Y. Girdhar and G. Dudek, "Optimal online data sampling or how to hire the best secretaries," in *Proc. IEEE Canadian Conf. on Computer and Robot Vision*, 2009, pp. 292–298.
- [24] M. Bateni, M. Hajiaghayi, and M. Zadimoghaddam, "Submodular secretary problem and extensions," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2010, pp. 39–52.
- [25] V. Shestak, E. K. Chong, A. A. Maciejewski, and H. J. Siegel, "Probabilistic resource allocation in heterogeneous distributed systems with random failures," *J. of Parallel and Distributed Computing*, vol. 72, no. 10, pp. 1186–1194, 2012.
- [26] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *Int. J. Robotics Research*, vol. 32, no. 12, pp. 1495–1512, 2013.
- [27] G. Hardy, J. Littlewood, and G. Polya, *Inequalities*. Cambridge University Press, 1934.
- [28] K. Museth, J. Lait, J. Johanson, J. Budsberg, R. Henderson, M. Alden, P. Cucka, D. Hill, and A. Pearce, "OpenVDB: An open-source data structure and toolkit for high-resolution volumes," in *Proc. Int. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2013.
- [29] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE Int. Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 1997, pp. 146–151.
- [30] G. Shmueli, T. P. Minka, J. B. Kadane, S. Borle, and P. Boatwright, "A useful distribution for fitting discrete data: revival of the conway–maxwell–poisson distribution," *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, vol. 54, no. 1, pp. 127–142, 2005.
- [31] A. J. Lee and S. H. Jacobson, "Sequential stochastic assignment under uncertainty: estimation and convergence," *Statistical Inference for Stochastic Processes*, vol. 14, no. 1, pp. 21–46, 2011.
- [32] S. C. Albright, "A Bayesian approach to a generalized house selling problem," *Management Science*, vol. 24, no. 4, pp. 432–440, 1977.