Multi-UAV Coverage Planning with Limited Endurance in Disaster Environment

Hongyu Song*,†, Jincheng Yu*,§, Jiantao Qiu*, Zhixiao Sun*, Kuijun Lang*, Qing Luo*, Yuan Shen§ and Yu Wang*

Abstract—For scenes such as floods and earthquakes, the disaster area is large, and rescue time is tight. Multi-UAV exploration is more efficient than a single UAV. Existing UAV exploration work is modeled as a Coverage Path Planning (CPP) task to achieve full coverage of the area in the presence of obstacles. However, the endurance capability of UAV is limited, and the rescue time is urgent. Thus, even using multiple UAVs cannot achieve complete disaster area coverage in time. Therefore, in this paper we propose a multi-Agent Endurance-limited CPP (MAEI-CPP) problem based on a priori heatmap of the disaster area, which requires the exploration of more valuable areas under limited energy. Furthermore, we propose a path planning algorithm for the MAEI-CPP problem, by ranking the possible disaster areas according to their importance through satellite or remote aerial images and completing path planning according to the importance level. Experimental results show that our proposed algorithm is at least twice as effective as the existing method in terms of search efficiency.

I. INTRODUCTION

For scenes such as floods and earthquakes, the disaster area is large, and rescue time is tight [1]. Multi-UAV exploration is more efficient than a single UAV [2]. [3] propose a multi-robot search and rescue approach in a disaster environment. The evaluation index is the number of rescued survivors and time to rescue a certain number of survivors. As the number of UAVs increases, the average time to rescue each survivor is significantly reduced.

Existing UAV exploration work models multi-UAV exploration as a coverage path planning (CPP) mission [4], where UAVs start from an arbitrary starting point in a space containing no-fly zones and obstacles. However, as the UAVs cruise at least 30–40 meters from the ground [5], there are few obstacles in the real-world rescue, leading to the inapplicability of this kind of CPP modeling.

However, the endurance capability of UAV is limited [6], and the rescue time is urgent. Thus, even using multiple UAVs cannot achieve complete disaster area coverage in time. Zhengzhou has flooded this year. Its Zhongyuan District is 193 square kilometers. It also takes about 6.9 hours. Taking into account the limited endurance, UAVs are needed to explore places first where dangers are more likely to occur. For example, the probability of a fire in a stadium is significantly greater than that of a lake. Therefore, we have defined a multi-Agent Endurance-limited CPP (MAEI-CPP) problem based on a priori heatmap [7] the disaster area, which requires exploring more valuable goals under limited energy. Thus, trapped people or disaster-stricken property will have a greater chance of being rescued and protected. Currently, it is necessary to use pre-disaster maps/satellite images to detect and divide valuable areas.

The optimal solution of covering as many important goals as possible within a specific range with endurance constrain is considered to be complicated [8]. Using multiple robots to cover the target is even more challenging. Simple path coverage algorithms, such as Zig-Zag [9] continuous search, do not distinguish between targets, and multiple robots are prone to path repetition. The naive greedy algorithm will fall into a local optimum and interfere with each other during multi-robot collaboration and cannot be directly used to solve the MAEI-CPP problem.

To address above problems, We propose a SVReC multi-robot exploration framework as illustrated in Fig. 1 with following contributions:

- A multi-Agent Endurance-limited CPP (MAEI-CPP) problem is formulated, suitable for various potentially disaster-affected environments.
- A method that can generate a priori heatmap of the pre-
disaster environment from an environmental satellite map or remote aerial map is proposed.

- Several path planning algorithms for MAEl-CPP are proposed. Among them, the SVReC algorithm has outstanding performance in search efficiency.

According to the problem we defined (Section III-A), we first convert the map obtained by aerial photography into heatmap (Section IV), and then we generate the optimal path (Section V) according to the priori heatmap. We set up experiments (Section VII) based on the above problems and algorithms. Based on the experiments, we obtain conclusions and think of future work (Section VII).

II. RELATED WORK

Coverage path planning (CPP) has been widely discussed in the robotics field. [10] is a classic work that proposes Backtracking Spiral Algorithm (BSA) in the coarse-grain grid map. This algorithm is able to perform a complete coverage in an environment with occupied cells. A non-repetitive closed-loop coverage is proposed by [11] in a closed environment containing obstacles on the basis of [12]. [13] expands [11]’s work and provides a solution that can reasonably allocate the load distribution [14] of multiple robots. [15] use a Zig-Zag-like path to complete the path coverage targeted at 3D reconstruction, which can work in both convex and non-convex environments. The modelling of these traditional CPP problems is oriented to cover the entire map. They are not suitable for the MAEl-CPP problem because the aim of MAEl-CPP is to maximize the search efficiency, not wholly cover the map, and the physical constraints make it challenging to cover the entire map.

The map used for CPP problem usually follows a certain distribution, in [10] the grid cells are divided into two types, covering and non-covering, to deal with path planning. [16] provides a strategy to distinguish interested and uninterested viewpoints. An et.al. proposed [17] based on [16], in which the prior distribution map for coverage is introduced. [18] use a probability map of Gaussian distribution and proposed an algorithm that can solve single-robot CPP problem in this kind of map. However, it is difficult for the above maps to describe the a priori conditions specific to a certain area when the disaster occurs. Because the range of the real-world disaster hazard does not deliberately follow a certain distribution, and these maps are also difficult to deploy specific search efficiency-oriented tasks. Therefore, a map that can be constructed according to the potential disaster risks of various real environments is necessary for the MAEl-CPP problem.

In some CPP algorithms, physical constraints are adopted. [3] limits the time of searching the survivors. [18] use the time step of CPP to limit energy consumption. Each waypoint will cost a time step. Similarly, [13] introduces c as the capacity of the robot, which is the summation of total nodes in spanning tree. [19] considers more peculiar features of the robot. UAV’s energy consumption limit needs to be rigorously stated, but different kinds of UAVs have different efficiencies and other parameters, so the energy consumption of MAEl-CPP needs to be easily extended to different UAVs. Thus in this paper, endurance is the total path length calculated from the Euclidean distance between waypoints.

III. PROBLEM DEFINITION & PRELIMINARIES

A. MAEl-CPP Problem

A new MAEl-CPP problem is defined in this section. We hope that the potential disaster level can be defined with representative features in the entire map. This kind of level should also be able to describe the probability of the occurrence of a certain disaster in chosen area. It also needs to conform to the common sense of search and rescue problems as much as possible. Only then can we deploy UAVs in a more realistic simulation environment or a real environment. Then according to these characteristics, the grid map of the task space will be converted into a heatmap. The heatmap is a concise and informative map that can shows the differences between cells. The detail about how to get a heatmap from a grid map will be introduced in Section V.

In our Problem, a graph structure [20] called Heat-Graph $H$, the vertices in Heat-Graph correspond to the cells in heatmap one-to-one and have the same positional relationship as shown in Fig. 2. For example, given cell $C_{xy}$ and vertex $\eta_{ij}$, whose vertex weight is called Efficiency weight $r$, and $i$ is the row and column in the graph structure. $r(\eta_{ij})$ is mapped from heat-value $h(C_{xy})$ in heatmap as shown in Fig. 2 (b).

As a physical limitation, we stipulate that the endurance (Maximum path length) of each UAV is $D$. The edge weight $e(\eta_{ij}, \eta_{kl})$ between neighbor vertices is shown in Fig. 2 (a). the size of the grid in heatmap is 40mx40m and thus

$$\eta_{ij} = G(C_{xy}), i = x/40, j = y/40 \quad (1)$$

$$e(\eta_{ij}, \eta_{kl}) = \sqrt{(i - k)^2 + (j - l)^2} \quad (2)$$

$$r(\eta_{ij}) = \begin{cases} 10 & h(C_{xy}) = 1.0 \\ 10h(C_{xy}) - 2 & \text{else} \end{cases} \quad (3)$$

with $G$ is the map from heatmap to Heat-Graph. And the max accumulated edge weight $d_m = D/40$. We map $h(C_{xy})$ to $r(\eta_{ij})$ with Equ. [3] because we want to make one class really important. The other classes should be different and must contain a class that is worthless to be covered.

Given $H$ and $n$ UAVs, then the flight trajectory of the $i$th UAV $U_i$ is $\Gamma_i = (G(\gamma_i))$, $\gamma_i$ is a list that contains $M$ $\eta$s, $R_i$ and $d_i$ are accumulated Eff-weight and accumulated edge weight. Then the solving process of path-graph $\gamma_i$ can be summarized as the following problem:

$$\arg \max_{\gamma_i} R_i, \quad s.t. i \leq n, d_i \leq d_m \quad (4)$$

$$R_i = \sum_{m=1}^{M} r(\eta_m), d_i = \sum_{m=1}^{M-1} e(\eta_m, \eta_{m+1}), \quad \eta_m \in \gamma_i \quad (5)$$
邻居的顶点

卫星图像 fig. 3 (a) of our simulation scene was taken during the search and rescue process. So a high-altitude satellite image was adopted to classify disaster classes. These maps can not be used in the search and rescue of the post-disaster environment. However, we will elaborate on the heatmap in Sec. IV. (c) Represents the eight neighbor vertices of a vertex, red is current vertex, green and blue are neighbor vertices of diagonal and translation respectively.

**IV. POTENTIAL DISASTER HEATMAP**

Our map should be generated efficiently and can give different Eff-weights (Section III-A) to the cells that contain different scenes, for example, the water scene and the house scene. Only by building a map based on the probability of the occurrence of the actual disaster in this cell, our map can be used for the actual MAEI-CPP problem. However, it is challenging to achieve a map like this. Because most current maps for the CPP problem only care about the grid occupancy [24] or the probability distribution as described in Section I, an image map from the real environment is not able to be converted to these kinds of maps, and these maps can not be used in the search and rescue of the post-disaster environment because they are lack of comprehension of the potential disaster probability of various scenarios and the dangerous level of them after the disaster.

The map also needs to guide the UAV’s low-altitude flight during the search and rescue process. So a high-altitude satellite image fig. 3 (a) of our simulation scene was taken and segmented into a grid map. The cells in the grid map are divided into five classes according to the possibility that people in the scene may be trapped or face danger when the fire occurs. These classifications are only based on common sense, and there can be other criteria for the evaluation of classes in the missions of other scenarios or other types of disasters. However, this classification idea can be universal.

After having an idea of generating the heatmap described above, the first challenge is classification. Large-area city maps often contain tens of thousands of cells that need to be classified. Considering the difficulty of manual labelling, we only manually add scene tags to 20% of the cell images from the grid map, and then we use pre-trained DenseNet121 [25] for feature extraction and SVM [26] for classification. OpenCV [27] is also adopted to improve classification accuracy. After classification, the class labels are converted into heat-values, and the grid map is changed into a heatmap, which is adopted because it can concisely and efficiently distinguish the importance of different cells. Fig. 3 (a) shows examples of the five classes and their heat-value in a heatmap. Fig. 3 (b) shows our heatmap.

**B. NAIVE GREEDY PATH COVERAGE FOR A SINGLE UAV**

After a brief description of the problem in this paper, we first introduce a simple solution. The greedy algorithm [21] is a classic method for solving NP-Hard problems [22]. We are inspired by these articles and come up with the most straightforward Naive Greedy (Na-Greedy) Path Coverage algorithm. The calculation steps are as follows, given a Heat-Graph $H$, a UAV depot vertex $\eta_s$ and max accumulated edge weight $d_{max}$, we can get $\gamma_i$, $R$, and cover rate $\sigma$ after path planning. $\sigma$ is the percentage of covered vertices in all vertices. The pseudo-code of Na-Greedy is shown in Fig. 4.

Although this algorithm can quickly solve some Heat-Graph in some cases. However, due to the hard constraints of the CLOSE set [23] at this time, the UAV path will fall into a local optimal solution (that is, all points around the UAV are in the CLOSE set, and the UAV can not continue to plan the path after jumping out of the loop). To solve this problem, we will propose Heu-Greedy and SVReC in Section V.

**V. OUR ALGORITHMS**

The Na-Greedy (Naive Greedy) algorithm is a processing method essentially based on vertices Eff-weight. It has a weight trap problem for single-UAV CPP, and there is a weight interference problem for multi-UAV CPP. These two problems will lead to the end of the coverage planning loop because UAV can not find the next target vertex. The simplest solution is to add heuristic weights (Heu-Greedy) as shown in Fig. 4. The pseudo-code is as follows, whenever a candidate for $\eta_{next}$ is in CLOSE, we will make a penalty of -10 for its $r$ so that it can cover a known vertex but without accumulating weight, which can prevent it from jumping out of the loop when known vertex cannot be covered, and it can also distinguish each covered vertices. However, it will reduce the efficiency of search due to repeated coverage, which is called dynamic Weight Traps (repeatedly beating between the covered vertices). Our algorithm overview with pseudo-code is in Fig. 4.

In addition, both Na-Greedy and Heu-Greedy have the problems of Weight Sparse and Weight Inference. Although these two problems will not stop coverage, they will lead to some invalid coverage. For the above four problems, we adopted SVeC (skip vertices coverage) and VwRR (vertex weighted rotation range) (Fig. 5).
weight resolution reinforcement) methods to solve them. At the same time, for the multi-robot situation, based on the dynamic characteristics of UAV, we no longer focus on the traditional load distribution but prefer to improve the overall search efficiency without interfering with each other. An overview of the above four problems and the correspondence between them and each part of the SVReC algorithm are shown in Fig. 5.

A. Weight Trap

Weight Trap occurs when all reachable eight-neighbourhood vertices of the current vertex have been covered. The reachable eight-neighbourhood vertices refers to the neighbor vertices that do not exceed the map range. The Weight Trap also takes place when we use heuristic weights as soft constraints. UAV can choose the vertices that have been covered as potential paths, but because the r of vertices alternately descends, and vertices alternately become the next vertex in path. As a result, the UAV has been doing ineffective coverage between these vertices. Weight Trap is shown in Fig. 5 Weight Trap (a).

SVeC is used to solve this kind of problem. And the pseudo-code is shown in Fig. 4. When the reachable eight-neighbourhood vertices of the current vertex are all covered, the next target \( r_{\text{next}} \) of UAV is specified as the nearest vertex to the current vertex whose r is greater than a certain threshold \( r_{th} \). The solution of Weight Trap is shown in Fig. 5 Weight Trap (b).

B. Weight Interference

Similar to Weight Trap, Weight Interference is because the vertices covered by other UAVs and current UAV. These vertices surround the current vertex of the current UAV during multi-UAV collaborative coverage. In turn, when this situation is encountered, the coverage stops or loops between the vertices that have been covered, as shown in Fig. 5 Weight Interference (a).

This kind of problem can also be solved by SVeC. It is different from A, when multiple drones are used, all drones share the same CLOSE during planning. So Weight Inference can be considered as a multi-UAV version of Weight Trap. The solution is shown in Fig. 5 Weight Interference (b).

C. Weight Sparse

When the important areas of our heatmap are not continuous but are separated by some non-important areas, Weight Sparse will occur. When the UAV is in an area where the average \( r \) of the uncovered reachable eight-neighbourhood vertices of the current vertex is lower than a certain threshold \( r_{th} \), then the local solutions are all not good enough. Weight Sparse will reduce the efficiency of search because UAV has less probability to do efficient coverage here. For example, it is unreasonable to search for drowning people on land and it is harmful to people who really need to be rescued in the water. As shown in Fig. 5 Weight Sparse (a).

Similarly, we can also use SVeC to solve this type of problem when the average \( r \) of the uncovered reachable eight-neighbourhood vertices of \( \eta_c \) is equal to 0 or lower than a particular minimum value \( r_{min} \). We designate UAV’s next target \( \eta_{\text{next}} \), which is the nearest vertex whose r is greater than a certain threshold \( r_{th} \). As shown in Fig. 5 Weight Sparse (b). In this way, the accumulated Eff-weight can be ensured as more as possible in the case of limited
endurance, and there is no need to consume UAV energy in areas where there is no relative important vertices.

D. Weight Redundancy

When the above three problems do not exist, there is another problem that may affect the quality of local solutions because UAV often faces several uncovered reachable vertices in the eight-neighborhood with the same \( r \) as shown in Fig. 5. Weight Redundancy (a). These solutions may be equivalent and redundant at the current step, but from the perspective of the overall MAEI-CPP task, they have apparent differences.

So we use VwRR to solve this kind of problem. The solution is shown in Fig. 5. Weight Redundancy (a), and the pseudo-code is shown in Fig. 4. The cells’ average heat-value and the corresponding vertices’ average Eff-weights in the green box are significantly higher than those in the blue and yellow boxes, so when \( \eta_{max} \) is specified as the green vertex \( C \), we have a higher Eff-weight in the next few coverage steps because potential efficiency is optimized.

VI. RESULTS & ANALYSIS

In this section, the performance of different algorithms while solving the MAEI-CPP problem in City Park [28] heatmap and random heatmaps will be analysed and discussed. City Park heatmap is generated from UE4 [29] using method in Section IV and random heatmaps are directly generated from Matlab using random numbers. At the same time, we will also set up control experiments on multi-UAV and single-UAV. Zig-Zag, Na-Greedy, Heu-Greedy, and SVReC algorithm proposed in this paper will be compared in cover-rate \( \sigma \), acumulated Eff-weight \( R \), and time consuming for algorithms to accumulate a certain \( R \).

A. Simulation Environment

For simulation, UE4 and Airsim [30] are adopted. We first load and edit the environment according to our task requirements in the UE4 engine and rasterize it into cells of \( 40m \times 40m \). After that, the heatmap is obtained by the method introduced in Section IV. For CPP part, we calculate the directed subgraph \( \gamma \) in the graph structure through our algorithm, then convert it into waypoints of UE4 coordinates. Waypoints are sent to the UAVs for position-based control, and then UAV performs waypoints tracking and hovers over each cell that needs to be searched. It has been verified that our method can generate a heatmap based on the simulation environment and our classifier’s accuracy is over 91%. UAVs can successfully perform path planning based on our calculation results.

We apply the four algorithms (Zig-Zag, Na Greedy, Heu Greedy, and SVReC) to the heatmap of the City Park simulation environment and simultaneously display the heatmap with coverage paths and the UAV’s downward-looking cameras and flight status in the simulation environment. This environment is more similar to the real environment than the random heatmaps. As shown in Fig. 6, we use three UAVs and set \( d_m = 270 \) for the simulation experiment. The SVReC algorithm obtained at the end of this article can cover most of the important areas in the simulation experiment.

B. Eff-weight Evaluation

In Eff-weight Evaluation, the experimental settings for the performance of the algorithms are shown in Tables I and II. We test the four algorithms in the two cases: \( n = 1 \) and \( n = 6 \). In the City Park heatmap, a single UAV is set to \( d_m = 90 \) and \( d_m = 150 \), and multiple UAVs are set to \( d_m = 180 \), \( d_m = 90 \). In order to further test the capabilities of our algorithm, we randomly generated some heatmaps. Although it does not have a more common-sense regional distribution like the simulation environment, it has a larger size and can also simulate some extreme situations. We have adopted 50x50 and 100x100 random heatmaps. If the grid-scale is the same as the grid map of the simulation environment, the 100x100 heatmap mapped to the real environment is 16 square kilometers. In these maps a single UAV is set to \( d_m = 250 \), \( d_m = 550 \), \( d_m = 1100 \) and \( d_m = 2100 \), multiple UAVs are set to \( d_m = 450 \), \( d_m = 1050 \), \( d_m = 1800 \) and \( d_m = 4200 \). In all maps, the endurance load of each UAV in the multi-UAV case is one-sixth of \( d_m \). Each round of experimental UAVs starts from a randomly generated depot vertex, and different algorithms use the same random Depot vertex in each round of experiments. After the coverage is over, we will record the coverage rate and the cumulative Eff-weight.

Comparison: As shown in the columns under the label City Park in Tables I and II when the map size is small, Heu-Greedy and SVReC perform better than Zig-Zag and Na-Greedy. SVReC adopts a logic that focuses more on potential trends and is more efficient in search. SVReC is obviously better than the other three algorithms. Although in some cases, the coverage rate of SVReC is slightly lower than that of the other three algorithms, it can still achieve higher search efficiency, which shows that it is reasonable for us to solve the Weight Redundancy and Weight Sparse problems. Because this behavior increases the efficiency of search and the probability of survivors being rescued, thereby loss of life and property is reduced. In random large size heatmaps, as shown in the columns under label 50x50 and 100x100 in Tables I and II. Na-Greedy algorithm is obviously at a disadvantage because the Weight Trap and Weight Inference we mentioned before will cause its coverage rate significantly reduced. Although Heu-Greedy faces these two problems
### TABLE I: Single UAV Experiment Result

| n=1 | Map Size | City Park 19x20 | 50x50 | 100x100 |
|-----|----------|----------------|-------|---------|
|     | d_m=90   | d_m=150       | d_m=250 | d_m=550 | d_m=1100 | d_m=2100 |
|     | σ ∑ R_i  | σ ∑ R_i       | σ ∑ R_i | σ ∑ R_i | σ ∑ R_i | σ ∑ R_i |
| Algorithm |       |               |         |         |         |         |
| Zig-Zag | 90      | 282           | 150     | 444     | 250      | 426      | 550      | 976      | 1100     | 2034     | 2100     | 3616 |
| Na-Greedy | 80      | 398           | 131     | 596     | 177      | 672      | 133      | 476      | 298      | 1200     | 397      | 1578 |
| Heu-Greedy | 80      | 398           | 131     | 660     | 204      | 760      | 439      | 1174     | 884      | 2612     | 1827     | 4182 |
| SVReC   | 96      | 436           | 121     | 698     | 181      | 988      | 379      | 1910     | 738      | 3976     | 1423     | 7210 |

### TABLE II: multi-UAV Experiment Result

| n=6 | Map Size | City Park 19x20 | 50x50 | 100x100 |
|-----|----------|----------------|-------|---------|
|     | d_m=180  | d_m=300       | d_m=450 | d_m=1050 | d_m=1800 | d_m=4200 |
|     | σ ∑ R_i  | σ ∑ R_i       | σ ∑ R_i | σ ∑ R_i | σ ∑ R_i | σ ∑ R_i |
| Algorithm |       |               |         |         |         |         |
| Zig-Zag | 220     | 662           | 260     | 826     | 650      | 1144     | 850      | 1362     | 1300     | 2112     | 1700     | 2758 |
| Na-Greedy | 159     | 664           | 256     | 974     | 372      | 916      | 649      | 1664     | 605      | 1478     | 575      | 1576 |
| Heu-Greedy | 155     | 646           | 230     | 898     | 332      | 902      | 798      | 1984     | 1077     | 2458     | 2440     | 5474 |
| SVReC   | 150     | 728           | 240     | 1104    | 323      | 1160     | 720      | 2506     | 1398     | 4438     | 2772     | 8924 |

### TABLE III: Time Consuming for Algorithms to accumulate a certain $R$

| n   | 1 | 6 |
|-----|---|---|
| R   | 200 | 400 | 600 | 800 | 200 | 400 | 600 | 800 |
| Zig-Zag | 254.3 | 644.0 | 933.3 | 1270.0 | 240.6 | 574.2 | 844.4 | 1076.7 |
| Na-Greedy | 157.6 | 405.6 | / | / | 53.9 | 85.6 | 124.2 | 170.1 |
| Heu-Greedy | 157.6 | 405.6 | 603.1 | 810.9 | 58.4 | 85.1 | 126.4 | 178 |
| SVReC   | 157.6 | 340.6 | 353.8 | 766.1 | 44.4 | 71.1 | 103.7 | 146.4 |

Moreover, multi-UAV is obviously more advantageous in shortening the search and rescue time, which can be seen in Section VI-C.

### C. Searching Time Evaluation

For searching time evaluation, we do a speed racing experiment in the City Park environment for further test. We record the average time in seconds for algorithms to accumulate a certain accumulated Eff-weight $R$ in single-UAV and multi-UAV situations, the speed of UAV is set to $15m/s$, for each cell center in trajectory we set a hovering time for 2 seconds, the result is shown in Table III.

**Comparison:** When we compare the last three columns with the third one in Table III, the result shows that Na-Greedy, Heu-Greedy and SVReC demand significantly less mission time for the same $R$ compared to Zig-Zag in both single-UAV and multi-UAV conditions. Nevertheless, single-UAV Na-Greedy cannot accumulates an Eff-weight more than 596 because of Weight Trap, Weight Sparse and Weight Redundancy as shown in the third and fourth columns of Na-Greedy. SVReC performs better than other algorithms in all situations, which proves that our definition of Eff-weight and SVReC’s internal logic is reasonable. When we compare the columns under the label 1 and 6, it is also worth noting that in the MAEl-CPP problem, multi-UAV can accumulate the

and will not cause the coverage to stop, it is easy to fall into Weight Trap where $r$ in the local optimal solution area alternately drops and invalid coverage is repeated between several vertices. When compared both Tables I and II as the map size increases, SVReC shows a more obvious advantage in $R$ compared to its performance in City Park map, which is significantly higher than the other three algorithms, and the coverage rate is also higher than that of Na-Greedy and Heu-Greedy. This is because as the map size increases, SVReC avoids various problems that may lead to a stop of coverage or invalid coverage. Although when endurances are close, the $R$ of multi-UAV SVReC is slightly lower than the single-UAV SVReC, the difficulty of designing a single UAV with such long endurance is obviously greater than that of assigning these load endurances to multiple UAVs. That is why we propose MAEl-CPP problem and algorithm.
same Eff-weight in a much shorter time than single-UAV.

VII. CONCLUSION & FUTURE WORK

This paper proposes a new MAEL-CPP problem with clear energy constraints and efficiency definitions, a heatmap that characterizes potential disaster risks, and several MAEL-CPP algorithms. After a series of analyses, our SVREC is significantly better than Na-Greedy, Heu-Greedy and Zig-Zag algorithms when the endurance is limited. And this algorithm also has the ability to deploy on large size maps.

Our algorithm also has certain limitations. For example, for this type of NP-Hard problem, the strategy of our algorithm cannot be guaranteed to be optimal. Secondly, our search and rescue method is currently based on an offline a priori environment and does not have online path planning capabilities if the UAV’s downward-looking camera is fixed.

We hope to deploy our algorithm on actual robots and actual terrain environments in the future. In addition, we also hope to combine Micro-Macro control on the UAV to implement more practical search and rescue actions.

REFERENCES

[1] H. Yang, R. Ruby, Q.-V. Pham, and K. Wu, “Aiding a disaster spot via multi-uav-based iot networks: Energy and mission completion time-aware trajectory optimization,” IEEE Internet of Things Journal, pp. 1–1, 2021.

[2] L. Ruan, J. Wang, J. Chen, Y. Xu, Y. Yang, H. Jiang, Y. Zhang, and Y. Xu, “Energy-efficient multi-uav coverage deployment in uav networks: A game-theoretic framework,” China Communications, vol. 15, no. 10, pp. 194–209, 2018.

[3] E. T. Alotaibi, S. S. Alqefari, and A. Koubaa, “Lsr: Multi-uav collaboration for search and rescue missions,” IEEE Access, vol. 7, pp. 58 817–58 832, 2019.

[4] M. Theile, H. Bayerlein, R. Nai, D. Gesbert, and M. Caccamo, “Uav coverage path planning under varying power constraints using deep reinforcement learning,” 10 2020, pp. 1444–1449.

[5] O. BALOTĂ, U. d. . A. i Medicină, V. Bucureti, and D. IORDAN, “Advanced uav lidar system for geospatial data collection.”

[6] M.-h. Hwang, H.-R. Cha, and S. Y. Jung, “Estimation for minimizing energy consumption of multirotor unmanned aerial vehicles,” Energies, vol. 11, no. 9, 2018. [Online]. Available: https://www.mdpi.com/1996-1073/11/9/2221

[7] Z. Cao, H. Gao, K. Mangalam, Q.-Z. Cai, M. Yu, and I. Malik, “Long-term human motion prediction with scene context,” in Computer Vision – ECCV 2020, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 387–404.

[8] N.-T. Nguyen and B.-H. Liu, “The mobile sensor deployment problem and the target coverage problem in mobile wireless sensor networks are np-hard,” IEEE Systems Journal, vol. 13, no. 2, pp. 1312–1315, 2019.

[9] A. Pintado and M. Santos, “A first approach to path planning coverage with multi-uavs,” in 15th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2020), A. Herrero, C. Cambra, D. Urdá, J. Sedano, H. Quintián, and E. Corchado, Eds. Cham: Springer International Publishing, 2021, pp. 667–677.

[10] B. Kalyanasundaram and K. R. Pruhs, “Constructing competitive tours from local information,” Theoretical Computer Science, vol. 130, no. 1, pp. 125–138, 1994.