Captain: Online Multi-Goal Adaptation for Autonomous Unmanned Systems via Optimal Control

Anonymous Author(s)

A APPENDIX

A.1 UUV Oceanic Surveillance

This example originates from [3, 4]. Rather than considering single-goal optimization for the UUV scenario in [3, 4], we extended it to achieve multiple dynamic goals under uncertainties and disturbance through elastic goal satisfaction, while other configurations like sensors are kept the same as [3]. The goals include:

- G_l: A segment of surface over a distance of $L_t = 100$ km is expected to be examined by the UUV within $\Delta = 10$ hours, while the distance threshold is $L = 90$ km.
- G_e: A total amount of energy $E_t = 5.4$ MJ is expected to be consumed, while the maximum amount of energy is $E = 6$ MJ.
- G_φ: The accuracy of sensor measurements is targeted at $A_t = 90\%$, while the accuracy threshold is set as $A = 80\%$.

The UUV is equiped with 5 sensors for ocean surveillance. The scanning time 10 hours is 360 time instance, $x_i, i \in [1, 5]$ is the portion of time the sensor i should be used during system operation in each instance. Acc_i is the accuracy of sensor i; E_i is the energy consumed by sensor; V_i is the scanning speed of sensor. q_i is portion of accuracy of sensor and p_i is for scanning speed respectively in decimals. The energy consumed is related with working accuracy and speed of sensor. The corresponding measures are listed as follows: $\mathcal{X}_l = \sum_{k=0}^{T}\sum_{i=0}^{N} x_i q_i V_i \tau$, $\mathcal{X}_e = \sum_{k=0}^{T}\sum_{i=0}^{N} x_i E_i \cdot \frac{e^{p_i+q_i}-1}{e^2-1}\tau$, and $\mathcal{X}_\varphi = \sum_{k=0}^{T}\sum_{i=0}^{N} x_i p_i Acc_i$, where $T = 360$, i.e., adaptations is performed every 100 surface measurements of the UUV state,, and the time instance k incremented by $1 \sim 100$. The goal satisfactions are listed as follows:

$$DS^2(\mathcal{X}_l) = \begin{cases} 1, & \mathcal{X}_l \geq L_t \\ \dfrac{\mathcal{X}_l - L}{L_t - L}, & L \leq \mathcal{X}_l < L_t \\ 0, & \mathcal{X}_l < L \end{cases}$$

$$DS^1(\mathcal{X}_e) = \begin{cases} 1, & \mathcal{X}_e \leq E_t \\ \dfrac{E - \mathcal{X}_e}{E - E_t}, & E_t < \mathcal{X}_e \leq E \\ 0, & \mathcal{X}_e > E \end{cases}$$

$$DS^2(\mathcal{X}_\varphi) = \begin{cases} 1, & \mathcal{X}_\varphi \geq A_t \\ \dfrac{\mathcal{X}_\varphi - A}{A_t - A}, & A \leq \mathcal{X}_{\varphi_k} < A_t \\ 0, & \mathcal{X}_\varphi < A \end{cases}$$

The Figure 1 shows the goal adaptation process of the UUV during operation with the case in [3]. As the capability of sensor reconfiguration increase, UUV could tune the sensor usage deliberately. At $k = 100$ we change the available energy change from 5.4 to 5.0 MJ, , at $k = 160$ we change the distance to be scanned from 100 to 105 km. The plots show that these changes in requirements lead to corresponding changes in the arrangement of sensor usage, as the time portion for $S2$ increases. Figure 1 also shows how Captain reacts to changes in sensor parameters and sensor failures. At $k = 220$, the measurement accuracy of sensor $S3$ drastically decreases from 83% to 43%, at $k = 290$, S4 stops working, and leads to switching the sensors to the optimal solution that $S1$ is more exploited. At last, the mission end with average measurement accuracy at 90.1%, scanning distance at 106.7 km, energy consumption at 4.98 MJ.

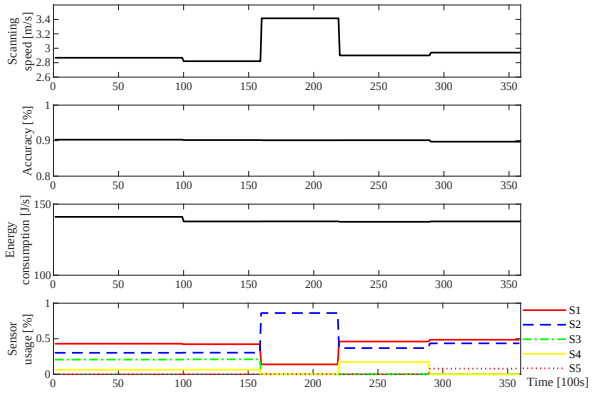


Figure 1: UUV adaptation optimal plannng results.

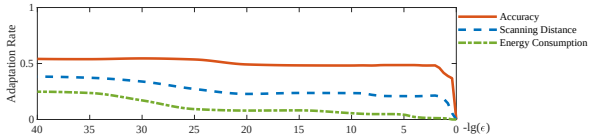


Figure 2: UUV case (# of incidents = 9)

In this scenario, the three methods are compared with their performance by adding random failures to parameters of sensors, i.e., sensor accuracy, scanning speed and energy consumption. For each method, we simulated the UUV with different number of random disturbances added to certain time instants. For each configuration, we ran the simulation for 500 rounds and computed the average accuracy, scanning distance and energy consumption. The results are shown in Table 1. Captain outperforms the other two strategies at the goal satisfaction overall and is more robustness when sensor failure increases. The variance of scanning distance and energy consumption of three strategies are compared in Figure 3(b) and Figure 3(c). The choice of different violation tolerance ϵ and the performance scanning accuracy achievement are illustrated in Figure 3(a). As the number of incidents increases, the impact of ϵ on goal satisfaction is more obvious.

As discussed in the paper, the choice violation tolerance ϵ determine which goal needs relaxation, while the larger the value is, the less number of goals in the *Violated Goal Set*, along with the underlying higher risk of no feasible solution. Thus the value of ϵ should be determined based on experimental data. Figure 2 illustrates adaptation rate of each soft goal in UUV case, as the increase of violation tolerance. In the UUV case, we choose $\epsilon_{\varphi,e,l} = \{10^{-3}, 0, 0\}$, when the adaptation rate of each goal tends to be gentle.

A.2 UAV Delivery Scenario

Then, we will illustrate how to apply goal satisfaction modeling into the case of UAV delivery system based on goals illustrated in the paper.

- Safety: The indicator to evaluate the safety requirement is the collision risk of UAV during the flight. Supposing that the obstacles detected by the UAV at time instant k is O_k, while the current state of UAV is s_k. Thus, the QM of safety is $\mathcal{X}_{S_{o,k}} = \frac{\|\boldsymbol{x}_k - \boldsymbol{x}_o\|_2 - r_a - r_o}{D_o}, \forall o \in O_k$. Such that the average distance between UAV and the center of obstacle reflects the safety risk.

$$DS^2(\mathcal{X}_{S_{o,k}}) = \begin{cases} 1, & \mathcal{X}_{S_{o,k}} \geq 1 \\ 0, & \mathcal{X}_{S_{o,k}} < 0 \\ \mathcal{X}_{S_{o,k}}, & otherwise \end{cases}$$

- Timelines: The total traveling time from time instant i to j is denoted as $\xi_{ij} = \sum_{k=i}^{j-1} \frac{\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2}{\boldsymbol{v}_k}$. The indicator of timeliness is $\mathcal{X}_\xi = \xi_{0T}$, the degree of satisfaction of the timeliness requirement of the whole trajectory DS_ξ is:

$$DS^1(\mathcal{X}_\xi) = \begin{cases} 1, & \mathcal{X}_\xi \leq \Delta_t \\ \dfrac{\Delta - \mathcal{X}_\xi}{\Delta - \Delta_t}, & \Delta_t < \mathcal{X}_\xi \leq \Delta \\ 0, & \mathcal{X}_\xi > \Delta \end{cases}$$

- Accuracy: The average quality of the information collected during the mission is denoted as $\mathcal{X}_\varphi = \frac{1}{\xi_{0T}} \sum_{k=0}^{T-1} \|\boldsymbol{\omega}\| \tau$, the degree of satisfaction is DS_φ is:

$$DS^2(\mathcal{X}_\varphi) = \begin{cases} 1, & \mathcal{X}_\varphi \geq A_t \\ \dfrac{\mathcal{X}_\varphi - A}{A_t - A}, & A \leq \mathcal{X}_\varphi < A_t \\ 0, & \mathcal{X}_\varphi < A \end{cases}$$

- Energy-saving: The total energy consumption from time instant i to j is denoted as $e_{ij} = \sum_{k=i}^{j-1} \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2 + \eta_1 \cdot \|\boldsymbol{v}_{k+1} - \boldsymbol{v}_k\|_2 + \eta_2 \cdot \|\boldsymbol{\omega}_k\| \tau$. The indicator of energy consumption is $\mathcal{X}_e = e_{0T}$, the degree of satisfaction of energy requirement DS_e is:

$$DS^1(\mathcal{X}_e) = \begin{cases} 1, & \mathcal{X}_e \leq E_t \\ \dfrac{E - \mathcal{X}_e}{E - E_t}, & E_t < \mathcal{X}_e \leq E \\ 0, & \mathcal{X}_e > E \end{cases}$$

A.2.1 Implementation in UAV delivery system. The whole process of applying goal adaptation method into UAV trajectory planning at each time instant k is shown in the algorithm 1.

Algorithm 1: Goal Adaptation in Trajectory Planning for UAV at time instant k

Input: Current state s_k, prediction horizon N, detected obstacles O_k, detected privacy regions C_k, and criteria ϕ for iteration stopping. Requirement specifications: preferred travel time Δ_t, travel time budget Δ, preferred energy consumption E_t, energy budget E, preferred information quality A_t, information quality budget A, preferred non-conflict distance D_o, D_c.

Output: Action plan $< \boldsymbol{v}_k, \boldsymbol{\omega}_k ... \boldsymbol{v}_{k+N}, \boldsymbol{\omega}_{k+N} >$

```
while φ = false do
    Initialize v_k...v_{k+N}, ω_k...ω_{k+N}, slash variables ε, δ;
    CurConstr ← ∅;
    for i ∈ {S, P, ξ, e, φ} do
        X_i ← Obs_k(s, G_i);
        Constr_i ← SlashConstr (g_i, ub_i, lb_i, δ_i, ε_i)
          CurConstr = CurConstr ∪ Constr_i;
    Construct Goal Satisfaction Checking problem P_1 with
     CurConstr;
    (ε*, δ*, v*, ω*) ← Solve(P_1);
    /*Solve the problem with Sequential Convex
     Programming*/;
    if (v*, ω*) ≠ ∅ then
        for i ∈ {S, P, ξ, e, φ} do
            VD_i ← (δ_i*, ε_i*);
        G_v, G_nv ←GoalAnalyze({VD_i, ε_i});
        for G_i ∈ G_nv do
            GoalConstr ← ConstrConstruct(s, g_i);
            CurConstr = CurConstr ∪ GoalConstr;
        for G_j ∈ G_v do
            HardConstr ← ConstrConstruct(s, lb_i, ub_i);
            HardConstr = HardConstr ∪ GoalConstr;
        Construct Goal Satisfaction Optimization problem
         P_2 with current G_v, CurConstr, HardConstr;
        (v_2*, ω_2*) ← Solve(P_2);
        if v_2*, ω_2* ≠ ∅ then
            (v*, ω*) ← (v_2*, ω_2*);
    if φ = true ∧ v*, ω* = ∅ then
        Report false and act some emergent actions, such as
         emergency landing;
```

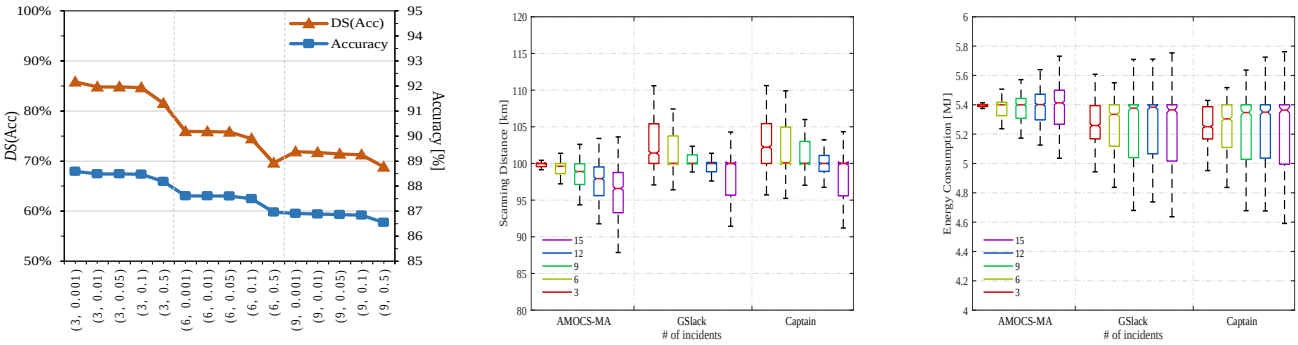
A.2.2 Robustness of `Captain`. In this section, we show the details of the experimental results in the UAV delivery scenarios discussed in the paper.

To test the robustness of `Captain` against the changes in the requirements and system behaviors, we randomly added several disturbances (up to 10% of target values) to the soft goals and system parameters at different time instants during the motion of the UAV. The simulation results are shown in Table 3, where the quantity of incidents denotes the adding of disturbances, and the values

Table 1: Goal satisfaction results of UUV case

Goals		*Accuracy* [%]			*Scanning Distance* [km]			*Energy Consumption* [MJ]			*Adaptation Rate*
Strategies		AMOCS-MA	GSlack	Captain	AMOCS-MA	GSlack	Captain	AMOCS-MA	GSlack	Captain	
# of incidents	3	88.26% 88.85	95.30% 89.53	**95.84%** **89.58**	96.64% 99.93	99.72% 103.91	**99.79%** **104.23**	99.03% 5.34	99.65% 5.24	**99.65%** **5.24**	34.33%
	6	74.84% 87.50	82.32% 88.24	**83.40%** **88.35**	89.38% 99.16	97.76% 103.38	**97.80%** **103.97**	97.89% 5.33	99.51% 5.19	**100%** **5.18**	55.33%
	9	57.78% 85.78	67.68% 86.77	**69.53%** **86.96**	79.03% 98.2	93.66% 103.04	**93.37%** **103.49**	96.14% 5.31	99.42% 5.16	**100%** **5.14**	71.67%
	12	43.77% 84.38	49.57% 84.96	**54.12%** **85.41**	64.86% 96.64	85.14% 100.50	**86.26%** **100.95**	92.29% 5.34	99.10% 5.17	**98.73%** **5.16**	80.83%
	15	24.05% 82.41	25.78% 82.58	**31.39%** **83.19**	44.15% 94.49	70.84% 98.81	**69.51%** **98.86**	90.28% 5.34	100% 5.18	**99.99%** **5.16**	80.87%

[a] The violation tolerance are $\epsilon_\varphi = 10^{-3}, \epsilon_e = 0, \epsilon_l = 0$



(a) Scanning accuracy with various incidents and ϵ. (Captain)

(b) Variance of scanning distance.

(c) Variance of energy consumption.

Figure 3: Scanning accuracy, distance and energy consumption with various probability of incidents and violation tolerance.

represent the degree of related goal's satisfaction, i.e., the values of $DS^j(\mathcal{X}_i)$. From Table 3, in the case of both requirement changes and system parameter changes, we can find that the increase of the frequency of introduced disturbances in the flight task can lead to the decrease of goal satisfaction degrees. However, Captain shows higher degrees of goal satisfaction and lower variance against AMOCS-MA and GSlack.

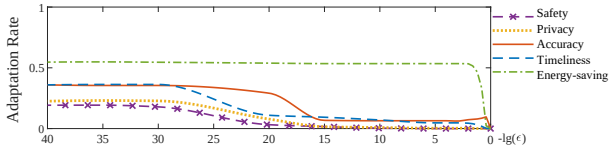


Figure 4: Goald adaptation rate in UAV case ($\rho_o = 2\%, \rho_p = 2\%$)

A.2.3 Scalability of Captain. To demonstrate the scalability of Captain with respect to different environments, we further simulated the UAV case on two selected real urban environments from the open building dataset of Portland in USA [1]. We used ArcGIS map to set up a 3D model based on the method in [2]. Their original

Table 2: Statistics on Overhead Data.

Case Study	*Strategies*	*Average* [s]	*Standard Deviation* [s]
UAV	AMOCS-MA	0.2039	0.0639
	GSlack	0.0991	0.1147
	Captain	***0.1301***	***0.1439***
UUV	AMOCS-MA	0.0423	0.0092
	GSlack	0.0180	0.0078
	Captain	***0.0344***	***0.0217***

spaces are $500\times500\times100m^3$ and $10^3\times10^3\times100\ m^3$, and compressed into $50\times50\times10\ m^3$ and $100\times100\times10\ m^3$, respectively. In the dataset, we can also obtain the longitude and latitude of the center of each building, as well as its average height and building types (i.e., industrial and commercial buildings, houses and apartments for living). The buildings for industrial and commercial use are viewed as obstacles, while houses and apartments are viewed as private regions. We used ArcGIS map to set up a 3D model according to the method introduced in [2]. As shown in Figure 5, the process of real urban scenarios modeling is explained. We set the range of longitude and latitude of working space on the ArcGIS map, and all the building are marked as blue (Figure 5(a)). Then the ArcGIS map

Table 3: Goal Satisfaction Results of the UAV with Dynamic Disturbances in Requirements and System Parameters

Goals			*Accuracy* (%)			*Traveling Time* (s)			*Energy Consumption* (unit)			*Adaptation Rate*
Strategies			AMOCS-MA	GSlack	Captain	AMOCS-MA	GSlack	Captain	AMOCS-MA	GSlack	Captain	
# of incidents	*Requirement Changes*	2	92.18% 89.39	97.18% 90.08	**97.36%** **90.27**	95.10% 13.96	98.20% 11.96	**98.85%** **11.44**	75.36% 23.70	86.21% 21.47	**86.82%** **21.35**	16.84%
		4	91.37% 89.20	95.95% 89.82	**96.35%** **90.11**	93.74% 13.51	97.94% 11.59	**98.37%** **11.15**	73.97% 23.44	84.47% 21.28	**85.04%** **21.17**	17.67%
		6	89.25% 89.14	94.33% 89.77	**95.09%** **89.95**	92.42% 13.57	97.00% 11.78	**98.15%** **11.22**	71.67% 23.64	81.89% 21.38	**82.95%** **21.23**	19.12%
	System Parameter Changes	2	87.49% 88.75	92.83% 89.53	**93.75%** **89.75**	98.55% 14.71	99.71% 12.83	**99.89%** **11.74**	75.21% 24.96	84.94% 22.98	**89.34%** **22.08**	19.07%
		4	84.24% 88.42	91.83% 89.32	**92.79%** **89.49**	97.08% 14.84	99.91% 12.80	**99.76%** **11.52**	73.03% 25.39	83.67% 23.26	**88.41%** **22.30**	21.40%
		6	78.15% 87.82	86.32% 88.79	**87.62%** **88.97**	96.12% 14.87	99.00% 12.70	**99.89%** **11.56**	72.30% 25.54	83.47% 23.30	**87.97%** **22.40**	26.28%

[a] The violation tolerance is $\epsilon_{SR,PR,\varphi,\xi,e} = \{10^{-20}, 10^{-20}, 10^{-10}, 10^{-20}, 0.005\}$ with environment setting of $\rho_o = 2\%, \rho_p = 2\%$

Table 4: Goal adaptation results for different scale of environment

Scale		*Accuracy* [%]	*Traveling Time* [s]	*Energy Consumption* [unit]	*Safety Risk*	*Privacy Risk*	*Real-time Performance*
50	Soft Goals	90	60	100	0	0	Adaptation Rate
	Hard Constraints	80	90	150	1	1	6/120
	Results	90	60.00	105.57	0	0	Overhead
	Satisfaction	100%	100%	88.85%	100%	100%	Avg. 0.066s Std. 0.139s
100	Soft Goals	90	90	200	0	0	Adaptation Rate
	Hard Constraints	80	150	300	1	1	3/203
	Results	90	101.50	210.94	0	0.1534	Overhead
	Satisfaction	100%	80.83%	89.06%	100%	99.98%	Avg. 0.132s Std. 0.180s

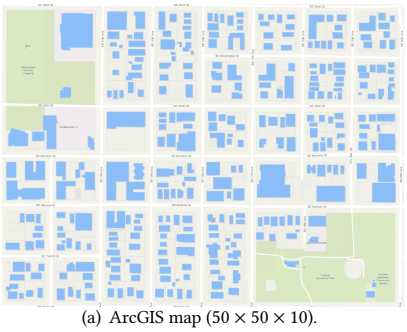
is turned into a binary map. Thus a 3D model of the urban scene is set up with the scale of $500m \times 500m \times 100m$ in Figure 5(c).

In each case, the flight task of UAV is to travel from the position $[0, 0, 0]$ to the destination $[49, 49, 0]$ and $[99, 99, 0]$ respectively, within the budget of accuracy, time and energy, as well as minor safety and privacy risk. The trajectory, state transition and goal achievement of the UAV at each time instant are shown in Figure 5(d), Figure 5(e) and Figure 5(f). So as to the scale of $1000m \times 1000m \times 100m$ in Figure 6. Table 4 summarizes the goal adaptation results in these two settings. we find that the states generated from `Captain` and the PD controller are almost the same, indicating that the planning results of `Captain` can be translated by the PD controller and executed by the UAV effectively.

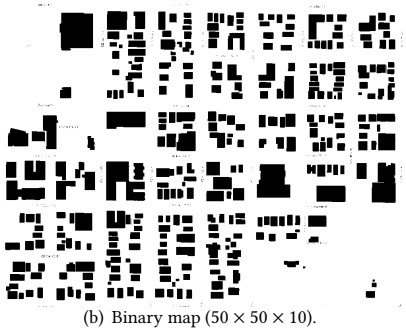
From our simulation results, in Setting 2, the flight task is completed in 60.07 s, consuming 106.27 units of energy without safety and privacy risk. In Setting 3, the task is finished in 101.5 s, consuming 210.94 units of energy. It does not have safety risk, but gets 4 points along the trajectory where the soft goal of privacy-preserving is violated. This results in an average 99.98% goal satisfaction along the path. `Captain` has high scalability as the two important steps *Goal Satisfaction Checking* and *Goal Satisfaction Optimization* are solved by SQP, which can handle large-scale optimization problems. In contrast, AMOCS-MA fails to compute a motion plan at real-time when the size of workspace increases.

A.2.4 Real-time performance of `Captain`. Finally, we analyze the cost to compute the optimal motion and sensor configuration plans for the given case studies. From Table 2, the gap of average computation time between AMOCS-MA and `Captain` is about 7 ms (0.0423 s versus 0.0344 s) in the UUV case, and 70 ms (0.2039 s versus 0.1301 s) in the UAV case. This is because AMOCS-MA considers the optimization of all goals while `Captain` only considers the determined violated goals. So AMOCS-MA requires more computations especially when there are more goals to be considered. `Captain` has larger variance since the number of goals to be optimized may vary at runtime.

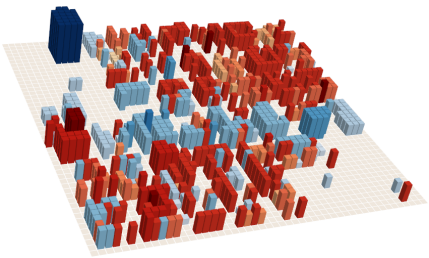
Fig. 4 illustrates adaptation rate of each soft goal in UAV case, as the increase of violation tolerance. In the experiments before, we choose $\epsilon_{S,P,\varphi,\xi,e} = \{10^{-20}, 10^{-20}, 10^{-10}, 10^{-20}, 0.005\}$ in the UAV case, when the adaptation rate of each goal tends to be gentle.



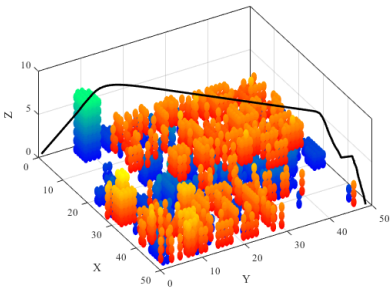
(a) ArcGIS map ($50 \times 50 \times 10$).



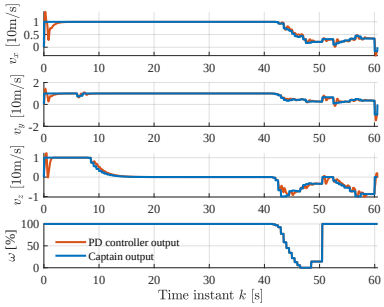
(b) Binary map ($50 \times 50 \times 10$).



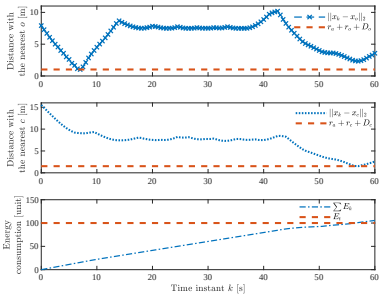
(c) Grid map ($50 \times 50 \times 10$).



(d) Flight Trajectory of UAV.

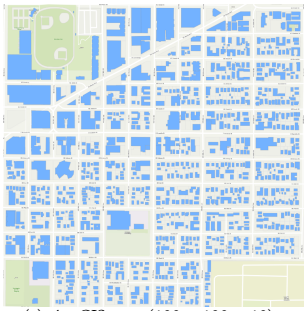


(e) State of UAV during the flight.



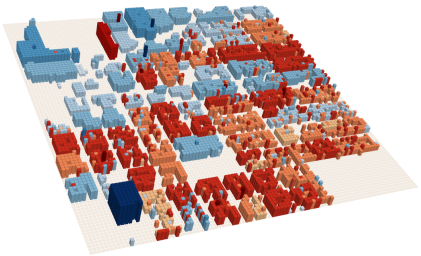
(f) Goal achievement of UAV during the flight.

Figure 5: Environment modeling with the scale of $500m \times 500m \times 100m$.

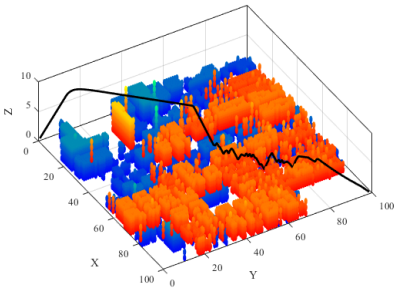


(a) ArcGIS map ($100 \times 100 \times 10$).

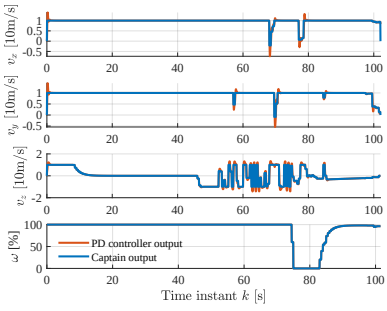
(b) Binary map ($100 \times 100 \times 10$).



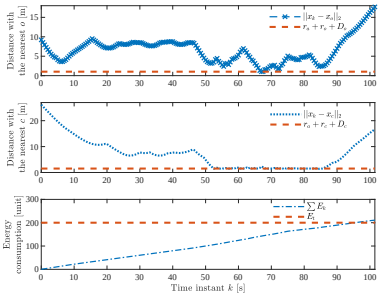
(c) Grid map ($100 \times 100 \times 10$).



(d) Flight Trajectory of UAV.



(e) State of UAV during the flight.



(f) Goal achievement of UAV during the flight.

Figure 6: Environment modeling with the scale of $1000m \times 1000m \times 100m$.

REFERENCES

[1] Steven J Burian, Srinivas Pradeep Velugubantla, K Chittineni, Sri Ram Kumar Maddula, and Michael J Brown. 2002. *Morphological analyses using 3D building databases: Portland, Oregon.* Technical Report. Utah. LA-UR, Los Alamos National Laboratory, Los Alamos, NM.

[2] Alejandro Hidalgo-Panagua, Miguel A Vega-Rodríguez, Joaquín Ferruz, and Nieves Pavón. 2017. Solving the multi-objective path planning problem in mobile robotics with a firefly-based approach. *Soft Computing* 21, 4 (2017), 949–964.

[3] Stepan Shevtsov and Danny Weyns. 2016. Keep it simplex: Satisfying multiple goals with guarantees in control-based self-adaptive systems. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. ACM, 229–241.

[4] Stepan Shevtsov, Danny Weyns, and Martina Maggio. 2019. SimCA*: A control-theoretic approach to handle uncertainty in self-adaptive systems with guarantees. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 13, 4 (2019), 17.