Appendix

1 Parameters for TVNN

Let $V = V_N \cup V_{\text{tar}} \cup V_S$, where each node $\bar{v} \in V$ has a time-varying position: $\mathbf{p}_{\bar{n}}(t) = \mathbf{x}_n(0)$ for $\bar{n} \in V_N$, $\mathbf{p}_{\bar{m}}(t) = \mathbf{y}_m(t)$ for $\bar{m} \in V_M$, $\mathbf{p}_{\bar{m}_i}(t) = \mathbf{y}_m^{\text{inter}}$ for $\bar{m}_i \in V_M^{\text{inter}}$, where $\mathbf{y}_m^{\text{inter}}$ is the position of the next intersection of m, and $\mathbf{p}_{\bar{s}}(t) = \mathbf{z}_s$ for $\bar{s} \in V_S$. The resource of interest is $\mathbf{R} \in \mathbb{R}^2$, where $R^{(1)}$ is time and $R^{(2)}$ is battery consumption. Each node $\bar{v}$ has time and battery constraints $[R_{v,\min}^{n,(i)}, R_{v,\max}^{n,(i)}]$ for $i = 1, 2$, depending on robot n. Specifically, for $\bar{v} \in V_N \cup V_S$, the time constraint is $[0, +\infty]$ and the battery constraint is $[0, b_n^{\max}]$. For $\bar{m} \in V_M$, the time constraint is $[0, \chi_m - \chi_m(0)]$, and the battery constraint is $[0, b_n^{\max} - \gamma_n d_S(\mathbf{p}_{\bar{m}}(t))/v_n]$. For $\bar{m}_i \in V_M^{\text{inter}}$, the time constraint is $[\chi_m^{\text{inter}}(0) - T_0, \chi_m^{\text{inter}}(0)]$, and the battery constraint is $[0, b_n^{\max} - \gamma_n d_S(\mathbf{p}_{\bar{m}_i}(t))/v_n]$.

As for connectivity, we define the set of reachable nodes $E_v \subset V$ for each node $v \in V$, i.e., $E_v = \{u \in V : (v, u) \in E\}$. Each edge $(v, u) \in E$ has an associated cost vector $\mathbf{C}(v, u) \in \mathbb{R}^2$. The connectivity and edge costs are summarized as follows.

$$
\begin{aligned}
C^{(1)}(\bar{v}, \bar{u}) &= d(\mathbf{p}_{\bar{v}}(t), \mathbf{p}_{\bar{u}}(t))/v_n + T_0 \cdot \mathbb{I}(\bar{v} \in V_{\text{tar}}), \\
C^{(2)}(\bar{v}, \bar{u}) &= \gamma_n C^{(1)}(\bar{v}, \bar{u}), \forall \bar{v} \in V \setminus V_S^{\text{dock}}, \bar{u} \in E_{\bar{v}}, \\
C^{(1)}(\bar{s}_0, \bar{s}_i) &= -C^{(2)}(\bar{s}_0, \bar{s}_i)/\beta_s, \\
C^{(2)}(\bar{s}_0, \bar{s}_i) &= -b_n^{\max} \cdot i/N_s, \forall \bar{s}_0 \in V_S^{\text{dock}}, \bar{s}_i \in E_{\bar{s}},
\end{aligned}
\tag{1}
$$

where $E_n = V \backslash V_N$, $\forall n \in V_N, E_{\bar{v}} = V \backslash V_N \cup \{\bar{v}\}$, $\bar{v} \in V_{\text{tar}}$; $E_{\bar{s}_0} = \bar{s} \backslash \bar{s}_0$, $\forall \bar{s}_0 \in V_S^{\text{dock}}, E_{\bar{s}_i} = V \backslash V_N \cup V_S^{\text{dock}}$, $\forall \bar{s}_i \in V_S^{\text{charge}}$; and the $\mathbb{I}(\cdot)$ is the indicator function.

2 Details of Martin's Process

In each iteration, the optimal l_c is selected from L_t^n, transferred into L_p^n, and propagated from $\bar{c}$ to all its viable nodes. The optimization objective is defined by the lexicographic order comparison function $\prec_{\text{lex}}$ (Line 2). This multidimensional comparison rule prioritizes lower time consumption while also minimizing battery usage. Notably, $\prec_{\text{lex}}$ satisfies the properties of *dominance* and *monotonicity*, ensuring that labels are extracted in the correct order [1]. The operation $\min^{\prec_{\text{lex}}}$ in Line 2 refers to the minimum selection based on the $\prec_{\text{lex}}$ ordering. The *propagate* process begins by generating a new resource vector through cost addition in Line 9. If the new vector is valid (Line 10), it is adjusted by $R_{v,\min}^{n,(i)}$ in Line 11 and create a new label. The right boundary of the resource window is a strict constraint that cannot be exceeded, while the left boundary, if not reached, is compensated, representing waiting time for $i = 1$

Algorithm 1: Martin's Process (MP)

Input: $G^n = (V^n, E^n), L_p^n, L_t^n$

Output: Multiobjective shortest path $\mathcal{A}$

while L_t^n *is not empty* **do**
  $l_{\bar{c}} \leftarrow \arg\min^{\prec_{\text{lex}}}_{l_{\bar{v}} \in L_t^n} \mathbf{R}_{l_{\bar{v}}}$
  $\bar{c} \leftarrow l_{\bar{c}}.\text{Node}()$
  if $\text{terminate}(l_{\bar{c}})$ **then**
    Return $\mathcal{A} \leftarrow \text{backtracking}(l_{\bar{c}})$
  $L_{\bar{c},t} \leftarrow L_{\bar{c},t} \setminus \{l_{\bar{c}}\}$
  $L_{\bar{c},p} \leftarrow L_{\bar{c},p} \cup \{l_{\bar{c}}\}$
  for $\bar{v} \in E_{\bar{c}}$ **do**
    `// Propagate to` $\bar{v}$
    $\tilde{\mathbf{R}}_{l_{\bar{v}}} \leftarrow \tilde{\mathbf{R}}_{l_{\bar{c}}} + \tilde{\mathbf{C}}(\bar{c}, \bar{v})$
    if $R_{l_{\bar{v}}}^{(i)} \leq R_{\bar{v},max}^{n,(i)}, \forall i \leq 2$ *and* $\hat{R}_{l_{\bar{v}}}^{(i)} \geq 0, \forall i \leq N_{tar}$ **then**
      $R_{l_{\bar{v}}}^{(i)} \leftarrow \max\{R_{l_{\bar{v}}}^{(i)}, R_{\bar{v},\text{min}}^{n,(i)}\}, \forall i \leq 2$
      $l_{\bar{v}} \leftarrow (\bar{v}, \tilde{\mathbf{R}}_{l_{\bar{v}}}, \bar{c})$
      if $\nexists\, l \in L_{\bar{v},p} \cup L_{\bar{v},t}$ *s.t.* $l \prec_P l_{\bar{v}}$ **then**
        $L_{\bar{v},t} \leftarrow L_{\bar{v},t} \setminus \{l \in L_{\bar{v},t} : l_{\bar{v}} \prec_{\text{P}} l\}$
        $L_{\bar{v},t} \leftarrow L_{\bar{v},t} \cup \{l_{\bar{v}}\}$
Return $\varnothing$

and invalid excess charge for $i = 2$. If the new label is non-dominated at $\bar{v}$, dominated labels in the temporary set are removed, and the new label is inserted. When the optimal label $l_{\bar{c}}$ meets the early termination condition, i.e., $\hat{\mathbf{R}}_{l_{\bar{c}}} = \mathbf{0}_{N_{\text{tar}}}$, the optimal task sequence $\mathcal{A}$ is obtained by backtracking each label's predecessor from $l_{\bar{c}}$.

3 Generalization

The proposed planning framework can be generalized in the following aspects: (I) *Robot Failures*. In the event of a robot failure, e.g., an unexpected UAV crash, this event is immediately treated as a triggering condition for online adaptation. Consequently, all targets assigned to the crashed UAV are added to V_l as the set of unassigned targets. A replanning is initiated to redistribute these tasks, via the same adaptation algorithm in *Online Execution*. (II) *Free Membership of Targets* In case new targets enter the workspace or existing targets leave the workspace, the proposed framework can still be applied. Namely, any target that leaves the workspace during execution is removed from the Node Network in real time, which automatically is not considered in the assignment. Similarly, when a new target enters the scene, it is detected by at least one robot and added to the set of targets to be assigned. Then, the same procedure can be applied.

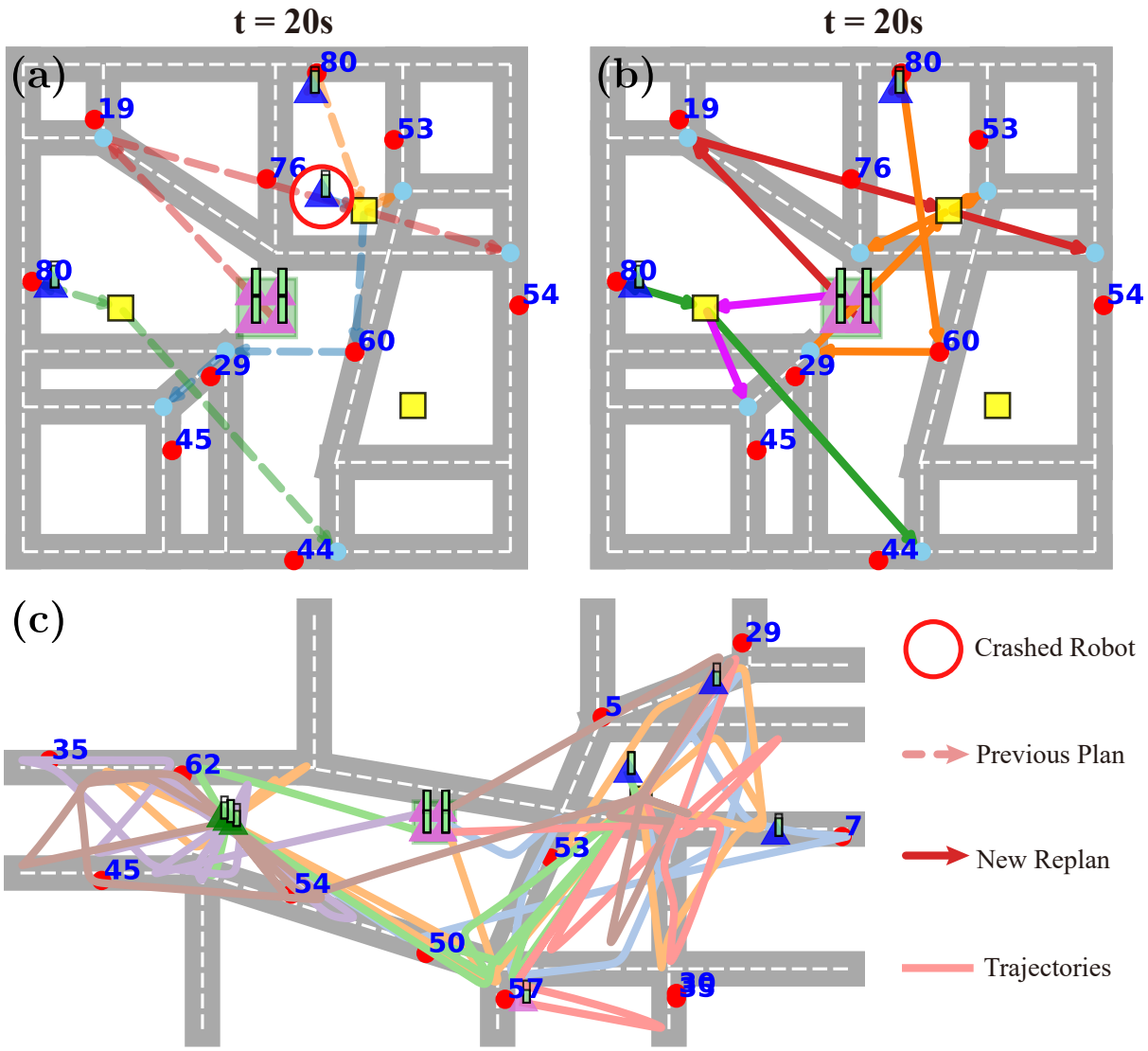

Figure 1: Illustration of the generalization circumstances. **(a)**: Previous plan result at $t = 20\,\mathrm{s}$ when a robot accidentally crashes. **(b)**: New replan after the robot crashed. **(c)**: Trajectories of 6 robots monitoring 15 targets that can dynamically enter and exit the road network over a period of $400\,\mathrm{s}$

Performances under generalized scenarios are also validated as shown in Fig. 1. Namely, one robot failed at $t = 20s$, after which the targets it was tracking are reassigned to other robots in the new replan. The adaptation takes $0.02s$ and the local plans of the other robots are modified accordingly. Moreover, 6 targets enter freely into the workspace during online execution, with the setup otherwise identical to Fig.3 in the paper. The trajectories of the employed robots are adapted to monitor the new targets.

Bibliography

[1] José Manuel Paixão and José Luis Santos. "Labeling methods for the general case of the multi-objective shortest path problem–a computational study". In: *Computational Intelligence and Decision Making: Trends and Applications*. Springer. 2013, pp. 489–502.