CoTFly: Making UAVs Think Where to Fly Next through Visual Chain-of-Thought Reasoning

Supplementary Material

7. Details of Data Collection

In this section, we present details of the UAV-CoT data production process. We first introduce the overall algorithm of data generation with a trajectory-aware sampling strategy in the form of pseudo-code. We then provide the full prompt used to guide the LLM in landmark planning, along with several examples illustrating the reasoning-chain completion supported by visual annotators. Finally, we summarize the statistics of the UAV-CoT training data.

Algorithm for the data collection pipeline. Algorithm 1 presents the pseudocode of the proposed trajectory-aware sampling algorithm used in the UAV-CoT data collection process to clearly explain the process of data generation, thereby facilitating understanding and reproduction.

Algorithm 1: Data collection with sampling

Input: Trajectories $\{T_j\}_{j=1}^{J}$ with lengths L_j, sampling rate r, trajectory weight α, and step decay factor λ.
Output: Sampled set $\mathcal{S}$.
Compute total step samples $N_{\text{tot}} = \sum_{j=1}^{J} L_j$ and global budget $B = \lfloor r N_{\text{tot}} \rfloor$;
Compute trajectory weights $u_j = L_j^{\alpha}, j = 1, \ldots, J$;
Set trajectory budgets $B_j = \left\lfloor \frac{u_j}{\sum_{k=1}^{J} u_k} B \right\rfloor$, $j = 1, \ldots, J$;
Initialize sampled set $\mathcal{S} \leftarrow \emptyset$;
foreach *trajectory* $T_j = \{(I, V_t, P_t)\}_{t=1}^{L_j}$ **do**
  for $t = 1$ **to** L_j **do**
    Compute normalized step index $p_t = \frac{t-1}{L_j - 1}$;
    Compute step weight $w_t = e^{-\lambda p_t}$;
  Normalize $\{w_t\}_{t=1}^{L_j}$ to a distribution $\tilde{w}_t = \frac{w_t}{\sum_{s=1}^{L_j} w_s}$;
  Randomly sample B_j step indices from $\{1, \ldots, L_j\}$ according to $\{\tilde{w}_t\}$ without replacement;
  Add the sampled steps of T_j into $\mathcal{S}$;
return $\mathcal{S}$;

Details of landmark planning and visual annotations. We design a comprehensive prompt that integrates the given information, including multi-view images (V_i) and the target-oriented navigation instruction (I). The prompt provides a deliberately detailed task requirement that encourages a structured chain-of-thought process, covering scene understanding, goal interpretation, candidate landmark identification, and the selection of a single best sub-goal candidate. In addition, a set of strict constraints (e.g., *NOT* conditions) is incorporated to guide the LLM (i.e., GPT-5 in this paper) in producing high-quality and most potential sub-goal landmarks. To avoid hallucinations, we output an empty sub-goal when no suitable landmarks are found. The final output format is simple and compact, consisting of only a short phrase that provides a more precise description for subsequent open-vocabulary visual annotation. We present the full prompt template in Figure 8. For the visual annotators, we leverage the currently best-performing open-vocabulary visual grounding tool, GroundingDINO, to acquire all visually grounded bounding boxes requested by the sub-goal description. For a clear description of the collection setting and results, we illustrate two examples of landmark planning results as well as the visual annotation results in Figure 9-14. For each navigation step, we show a certain view from five views.

Statistics of the UAV-CoT dataset. We construct the UAV-CoT dataset based on the UAV-Need-Help training set [27]. A detailed analysis of landmark semantics, the sampling distribution across different trajectory lengths and different navigation step indices, as well as the size and spatial distribution of visually grounded bounding boxes, has been provided in §3.3. In this section, we further report the overall statistics of the UAV-CoT dataset in Table 5, including the total number of trajectories (N_{traj}), the number of navigation steps (N_{step}), the amount of CoT data generated under a 20% sampling rate (N_{CoT}), the size of the sub-goal vocabulary (N_{vocab}), the number of valid sub-goals paired with bounding boxes ($N_{\texttt{sub-goal}}$), and the average length of the sub-goal descriptions ($Avg.L_{\texttt{sub-goal}}$). We find that the average length of the sub-goal descriptions (i.e., 1.81) is quite short, leading to a compact CoT. Specifically, one-word sub-goals account for 38.1%, two-word sub-goals account for 46.3%, and the longest sub-goal contains six words, representing only 0.03% of all sub-goals. The total number of trajectories and navigation steps is identical to that of UAV-Need-Help, among which 20% are annotated

You are an expert AI navigation planner. Your task is to identify the best intermediate navigation subgoal from **Final Navigation Goal** and **Images**.

GIVEN INFORMATION:

- **Images**: Five drone-view perspectives (front, left, right, rear, down).
- **Final Navigation Goa**l: {goal}.

First, think step-by-step to solve the problem. Structure your reasoning using the following thought process. This reasoning will not be in your final output.

CHAIN OF THOUGHT PROCESS:

1. **Scene Comprehension:** Based on all five images, give the description: what is the overall environment? (e.g., "This is a dense urban area with mixed residential and commercial buildings.")
2. **Goal Analysis:** Where is the **Final Navigation Goal** likely located relative to the drone's current position? Is it visible? Which direction seems most promising?
3. **Candidate Identification**: From all drone views, list 2-3 distinct, concrete, and relevant **landmarks** that could serve as potential subgoals.
4. **Evaluation & Selection**: Evaluate each candidate.
    - The landmark must be clearly visible in at least one image.
    - Does it move the drone closer to the final goal?
    - Is it a clear and unambiguous landmark?
    - Does it comply with all HARD RULES?
    - Select the single best candidate based on this evaluation.

After completing your internal reasoning, provide ONLY the final, one-line answer in the required format.

HARD RULES (must be followed):

- If the final navigation goal is "bridge", you must **NOT** output "bridge", "bridges", or similar paraphrases.
- The subgoal must be a minimal noun phrase (e.g., traffic light, etc). **Do NOT** provide a detailed description (e.g., traffic light on the right, etc).
- **If there is no suitable and visible subgoal, output exactly:** Subgoal: "".

FINAL OUTPUT FORMAT (one short line only):

Subgoal: <short phrase or "">

Figure 8. The complete guiding prompt used in our data collection pipeline instructs the LLM to plan potential sub-goal landmarks. It consists of four main components: (1) the provided information, including images and instructions, (2) the task specification requiring a chain-of-thought reasoning process, (3) a set of strict constraints to ensure high-quality sub-goal descriptions, and (4) a final output format restricted to a single concise phrase or an empty response.

with CoT in UAV-CoT. All navigation steps in UAV-CoT have a continuous UAV pose previously annotated by human expert operators in [27], i.e., 6-DoF UAV attitude.

To comprehensively evaluate the reasoning capability of `CoTFly` as described in §5.2, we additionally collect 1000 CoT samples based on the seen test set of UAV-Need-Help, each containing a valid sub-goal paired with its corresponding bounding boxes.

8. Details of `CoTFly` Training

Table 7 lists the training details of `CoTFly`.

Pseudo-labeling training. Since we produce only 20% CoT training data via the proposed trajectory-aware sampling strategy for efficiency, the remaining 80% of the training data lacks annotated CoT labels and can only supervise action prediction (i.e., unbalanced training objectives), which leads to a noticeable degradation in the `CoTFly` 's CoT reasoning quality. To mitigate the issue, we adopt a classical pseudo-labeling training strategy [16]. We train `CoTFly` using the 20% annotated CoT data together with the AerialVG dataset in the first training round as described in §4. We then employ the trained round-one model to generate the sub-goals for the unlabeled 80% training data, which are treated as pseudo-labels. Finally, we combine

Dataset	N_{traj}	N_{step}	N_{CoT}	N_{vocab}	$N_{\texttt{sub-goal}}$	$Avg.L_{\texttt{sub-goal}}$
UAV-CoT	7922	427933	85587	1027	82566	1.81

Table 5. Additional statistics of the UAV-CoT dataset.

Method	Assistant	Full				Easy				Hard			
		NE(m)↓	SR(%)↑	OSR(%)↑	SPL↑	NE(m)↓	SR(%)↑	OSR(%)↑	SPL↑	NE(m)↓	SR(%)↑	OSR(%)↑	SPL↑
Travel	L1	130.60	11.40	31.13	10.45	96.27	12.47	33.31	11.29	167.49	10.62	28.91	9.81
CoTFly	L1	120.76	**19.41**	31.95	**16.47**	89.86	**16.75**	31.04	13.75	149.87	**21.91**	32.80	**19.02**
CoTFly	Visual	**119.48**	17.96	**33.96**	15.70	**88.73**	16.49	**34.94**	**13.83**	**148.47**	19.34	**33.05**	17.47

Table 6. Compare CoTFly with the previous state-of-the-art UAV VLN on the unseen test set of UAV-Need-Help on NE, SR, OSR, and SPL metrics. Following [27], we divide trajectories into two types: length less than 250m as easy and length greater than 250m as hard.

Parameters	**Round#1**	**Round#2**
Hardware environment	0.5 A100×days	56 A100×days
Objective	Next token prediction	Next token prediction Mean absolute error Cosine similarity loss
Images	0.04TB	0.67TB
Batch size	256	1024
Iterations	378	6686
Optimizer	AdamW	AdamW
Learning rate	5e-5	5e-5
Warm-up steps	11	201
Trainable weights	460M	469M
Trainable weights ratio	6.35%	6.46%
LoRA rank	64	64
LoRA alpha	16	16

Table 7. Training details of the two training rounds in CoTFly.

the pseudo-labeled data with the original annotated data and jointly supervise CoT generation and end-to-end imitation learning of actions in the second round.

9. Additional Evaluation Resuls

We have presented CoTFly's navigation performance on the seen test set of UAV-Need-Help in Table 1. Following [27], we further provide the navigation results on a more challenging scenario, i.e., the unseen test set of UAV-Need-Help, whose targets and scenes are never seen in the train set. The results are shown in Table 6. The total number of trajectories in the unseen test set is 1587. We find that CoTFly, equipped with either the L1 assistant or the visual assistant, achieves superior performance compared with the baseline, Travel. For example, CoTFly improves the success rate by +8.01% with the L1 assistant and +6.56% with the visual assistant, while reducing navigation error by 9.84 meters and 11.12 meters, respectively.

10. Visualization Results

We have provided one long-horizon navigation example of CoTFly with a 354-meter trajectory length in Figure 6. We give more visualization results in Figure 15-16.

11. Limitation and Discussion

Although we have developed a robust automatic data collection framework that engages advanced LLMs and reliable visual annotators to help VLN incorporate visual chain-of-thought reasoning, several limitations remain. First, the diversity of sub-goal landmarks is limited due to the narrow range of objects and scenes covered by existing UAV VLN datasets, and occasional inaccuracies in visual annotations can result in negative navigation trajectories. We try to mitigate the limited diversity and annotation inaccuracies by introducing a manually annotated high-quality visual grounding dataset, AerialVG, during training. Second, incorporating explicit visual chain-of-thought reasoning with bounding boxes will slow down the navigation inference. This can be optimized through LLM-efficiency techniques such as quantization, sparsification, efficient caching, etc. Furthermore, navigation speed can be improved by adaptive chain-of-thought reasoning only at selected intermediate points rather than at every step, allowing UAV agents to adaptively balance efficiency and navigation success rate.

This work presents a UAV-specific visual chain-of-thought reasoning method that introduces midway reasoning ability to tackle challenging long-horizon navigation tasks, proposes a data collection framework that leverages LLMs and visual tools as reliable annotators, and develops a two-round training mechanism to build a large VLM architecture compatible with both reasoning and acting in UAV-based VLN. We expect this work to bring three benefits to the community. First, the proposed visual reasoning mechanism may push the progress of improving the reasoning capability of UAV agents in solving complex navigation tasks. Second, the introduced data production framework may be applied to widespread training scenarios to promote the development of current data-driven machine learning. Third, we hope that the visual-assisted architecture will be helpful for current vision-based navigation systems.

Trajectory ID: 399a51e1-d9f5-48db-9e50-2c2cff197f38.

I: The red car is parked on a city street with a zebra crossing nearby, flanked by tall buildings with prominent advertisements, including one for SocaSola and an American flag, and is located near scaffolding with a green bike lane adjacent.

The number of navigation steps: 54.

Sampling hyperparameters (trajectory weight) α=0.6.

Sampling hyperparameters (step decay factor) λ=2.

Figure 9. (#1-1) Two examples show the configuration, inputs, outputs of the LLM planner, and visual annotation.

Figure 10. (#1-2) Two examples show the configuration, inputs, outputs of the LLM planner, and visual annotation.

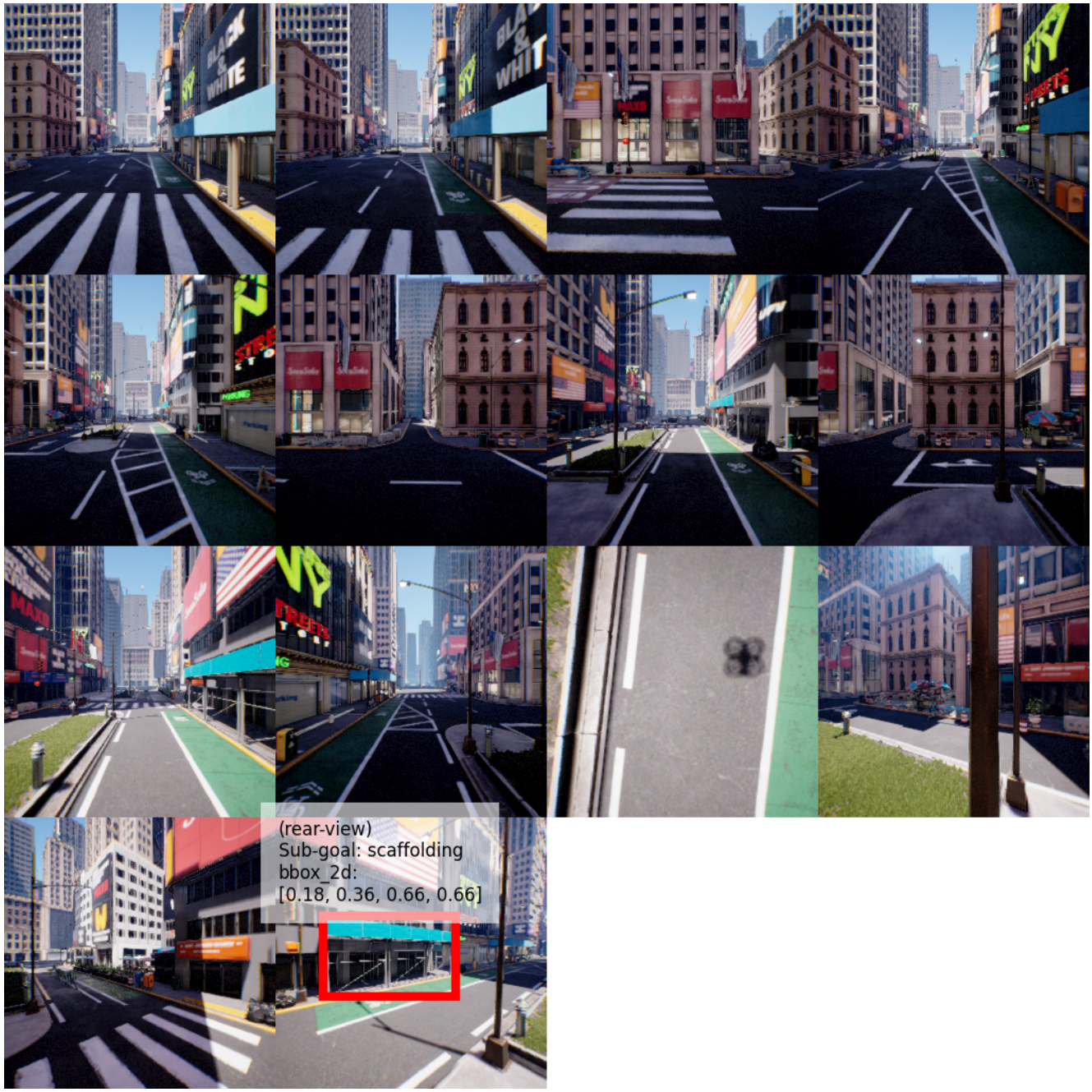

Figure 11. (#1-3) Two examples show the configuration, inputs, outputs of the LLM planner, and visual annotation.

Trajectory ID: 7a67bd27-3ce2-424a-ad74-c6a5f9666bd3

I: The brown car is positioned on a street with a bike lane, surrounded by tall buildings and a sidewalk featuring a canopy and planters with green foliage, accompanied by an advertisement for \"Black & White.\".

The number of navigation steps: 49.

Sampling hyperparameters (trajectory weight) α=0.6.

Sampling hyperparameters (step decay factor) λ=2.

Figure 12. (#2-1) Two examples show the configuration, inputs, outputs of the LLM planner, and visual annotation.

Figure 13. (#2-2) Two examples show the configuration, inputs, outputs of the LLM planner, and visual annotation.

Figure 14. (#2-3) Two examples show the configuration, inputs, outputs of the LLM planner, and visual annotation.

Figure 15. (#1-1) An additional visualization navigation example of `CoTFly` from the start point to the final target. We show the generated sub-goals and bounding boxes for each step.

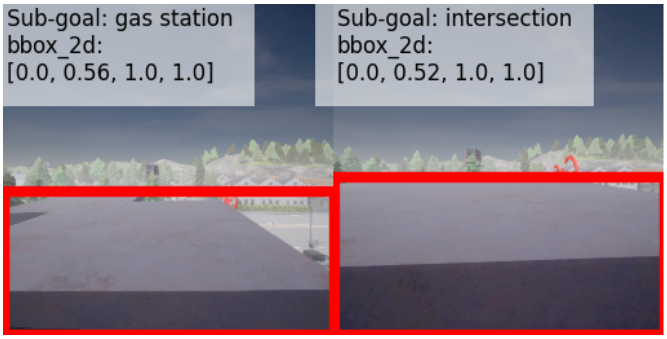

Figure 16. (#1-2) An additional visualization navigation example of CoTFly from the start point to the final target. We show the generated sub-goals and bounding boxes for each step.