End-to-End LLM Flight Planning with RAG-based Memory and Multi-modal Coach Agent

Amin Tabrizian [*1] **Arsyi Aziz** [*1] **Aarifah Ullah** [1] **Mahyar Ghazanfari** [1] **Pouria Razzaghi** [2] **Peng Wei** [1]

Abstract

Bridging the gap between human pilot intent and autonomous flight operation is critical for real-world electric vertical takeoff and landing (eVTOL) aircraft deployment. Flight planning traditionally relies on classic algorithms that struggle to incorporate flexible human preferences. We present **FRAMe**, an End-to-End Large Language Model (LLM) **F**light Planning tool with **RA**G-based Memory and **M**ulti-modal Coach Agent. Our system integrates a planner LLM with a multi-modal coach agent and retrieval augmented generation (RAG)-based memory to generate flight plans that satisfy mission constraints while aligning with human flight operator preferences. We demonstrate the system in a range of real-world-inspired scenarios of varying difficulty levels. Across four LLMs, the full FRAMe system (RAG and coach) yields the highest validity for every planner (up to 93.8% aggregate, 99% on Easy scenarios for the strongest planner) and shifts preference-relevant metrics in the operator-favored direction where the metric has headroom. FRAMe signifies how advanced LLMs can be deployed for human-centric mission planning, translating natural language instructions into safe, efficient, and flexible flight routes. The code is available at: github.com/amin-tabrizian/FlightPlanningLLMs

1. Introduction

Advanced Air Mobility (AAM) operations are projected to grow significantly in the near future, with unmanned aircraft systems (UAS) and eVTOL vehicles performing large volumes of flights for both cargo delivery and passenger transportation. Maintaining safe and efficient flight planning within such dense and complex airspace presents a major challenge. Flight restrictions, coupled with the often subjective and context-dependent nature of mission-specific requirements, make it difficult to define rigid, mathematical objectives that fully capture operational intent.

Classical path planning algorithms, such as A* and rapidly-exploring random trees (RRT, RRT*) (Kuffner & LaValle, 2000; LaValle, 2006), can compute optimal, obstacle-free trajectories; however, they depend on explicitly defined mathematical objectives and constraints, which limits their ability to capture nuanced human pilot preferences, such as trading off flight duration against waypoint complexity. As the scale and complexity of eVTOL operations continue to grow, there is an increasing need for automated planning systems capable of interpreting such nuanced mission intent.

One promising technology to bridge this gap is to use LLMs for planning. For instance, in robotics and navigation, the SayCan framework (ichter et al., 2023) demonstrated the capability of LLMs to guide robots using natural language to do feasible and contextually appropriate actions. In planning, hybrid approaches have leveraged LLMs to generate subgoals for classical planners (Meng et al., 2024; Liu et al., 2023; Dagan et al., 2024), while other methods integrate solver heuristics to better guide LLM generated plans (Wu & Mitra, 2024; Hirsch et al., 2024). Researchers have also explored combining LLMs with reinforcement learning (RL) to enhance reasoning capabilities. For instance, S2RCQL (Deng et al., 2025) augments prompts using information derived through Q-learning, and another approach employs LLMs to generate semantic hints for contextual RL in motion planning (Chen et al., 2026). Closer to our setting, recent work has brought LLMs directly into unmanned aerial vehicle (UAV) control: *TypeFly* (Chen et al., 2025) translates natural language into executable drone missions, and *GSCE* (Wang et al., 2025) introduces a structured prompt framework that improves the reliability of LLM-issued UAV commands.

Moreover, LLMs have also shown strong potential as proxies for human preference, which can be utilized to readily evaluate preference alignment (Zheng et al., 2023),

[*]Equal contribution [1]George Washington University, Washington, DC, USA 20052 [2]Metis Solutions Technology Inc, NASA Ames Research Center, Moffett Field, CA, USA 94035. Correspondence to: Amin Tabrizian <amin_tabrizian@gwu.edu>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

and recent work on *introspective planning* (Liang et al., 2024) shows that aligning an LLM's uncertainty with task ambiguity yields safer downstream plans—motivating the coach/verifier agent we adopt in this work. A second complementary direction is retrieval-augmented planning, where an embodied agent conditions on trajectories or experiences retrieved from a memory store: P-RAG (Xu et al., 2024) introduces progressive retrieval for everyday planning tasks, and STRAP (Memmel et al., 2025) retrieves sub-trajectories for robot policy learning. FRAMe adopts the same intuition for eVTOL route planning, retrieving preference-conditioned past flight plans rather than learning a policy from scratch.

Although LLMs have been explored extensively across these domains, their potential for real-world flight planning remains largely unexplored. To address this gap, prior work introduced an approach in which an LLM selected the route most aligned with human preference from multiple candidates (Tabrizian et al., 2024). That line of research was subsequently extended to an end-to-end flight planning generation system for eVTOL operations, employing various different chain-of-thought (CoT) prompting strategies (Wei et al., 2022; Tabrizian et al., 2025).

In this paper, we continue this progression by presenting an end-to-end LLM-based flight planning tool that incorporates a multi-modal coach agent and a memory system based on RAG. Specifically, we introduce FRAMe, a flight planning tool that enables human operators to generate eVTOL flight plans based on natural language preferences.

We propose a flight planning system that integrates a planner LLM with a multimodal coach agent and a RAG memory module. The LLM serves as the route planner, transforming textual prompts into sequences of geographic waypoints. The RAG module grounds the LLM's reasoning by supplying relevant context from prior planning experiences and their evaluations. The coach agent then validates and assesses each generated plan through a three-stage review: first, geometric tools verify that the flight plan is physically valid; second, the agent evaluates alignment with operator preferences by inspecting the rendered flight plan image; and third, the operator can optionally provide additional feedback. The resulting record is stored in the memory database, progressively improving future RAG retrievals. Our key contributions are as follows:

1. **FRAMe System:** We design a flight planning framework that integrates a planner LLM with a RAG-based memory module and a multi-modal coach agent, enabling natural language-driven route generation that respects no-fly zones and aligns with operator preferences.

2. **Annotator-Free Preference Evaluation:** We propose a quantitative evaluation framework built on three measurable preference objectives: minimizing flight distance, minimizing waypoint count, and maximizing polygon clearance. Because each objective is computed geometrically, preference alignment can be assessed objectively without human annotators. We pair this with a two-phase protocol, a warmup phase that seeds the memory followed by a read-only ablation phase, which isolates the contribution of retrieval augmentation.

3. **Multi-Model Ablation:** We conduct a systematic evaluation across four LLMs (OpenAI's o3-mini, o4-mini, GPT-5.4 (OpenAI, 2026), and DeepSeek-R1 (Guo et al., 2025)) and three scenario difficulty levels, benchmarking against a classical A* baseline. The ablation isolates each component: the coach provides a non-redundant validity lift on top of retrieval for every planner—most visibly on o3-mini, where RAG alone does not improve over Baseline.

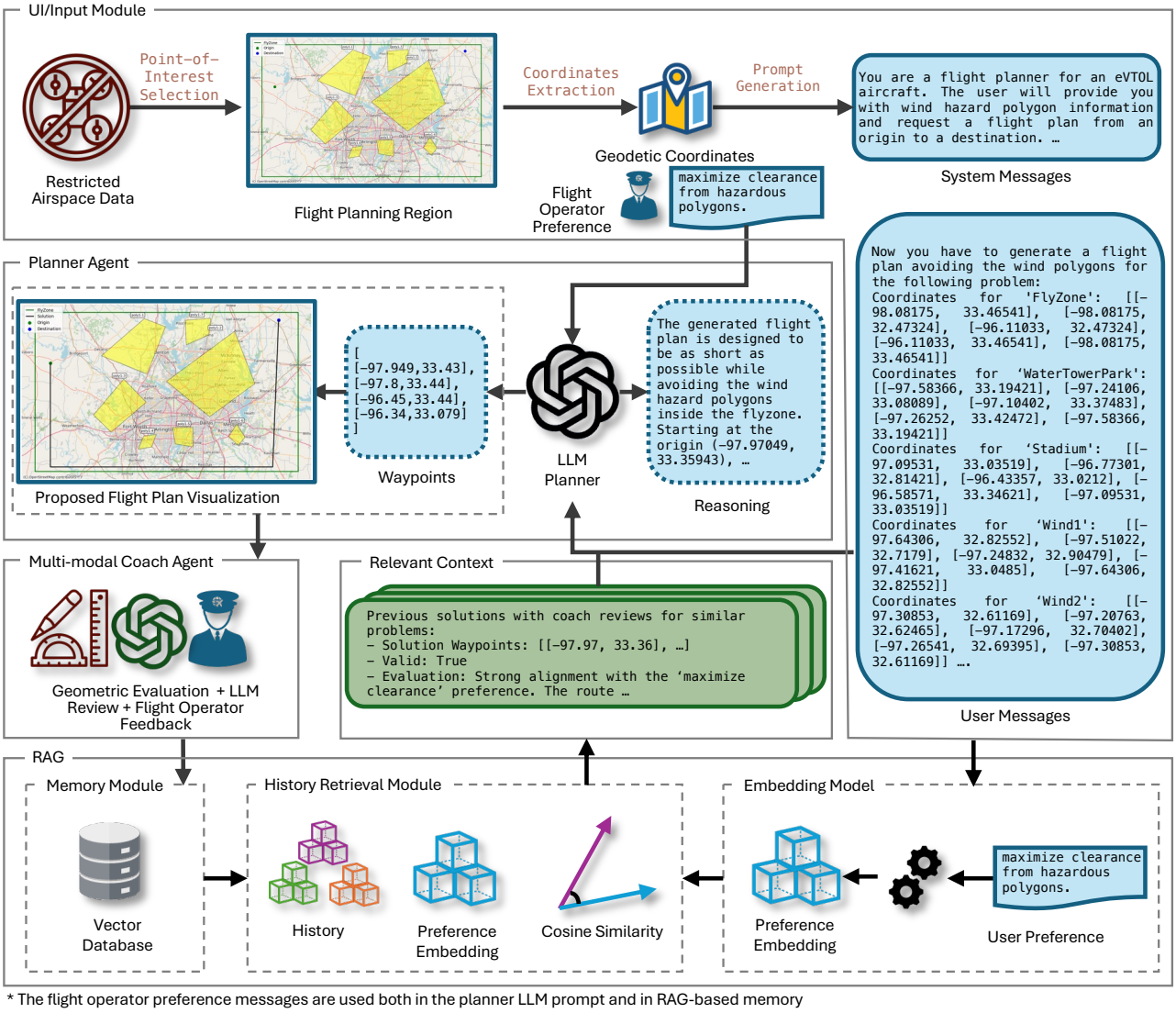
The remainder of this paper is as follows. Section 2 describes the FRAMe system architecture, including the planner agent, multi-modal coach agent, and RAG-based memory module. Section 3 presents the web prototype and experimental evaluation, covering plan validity, preference-metric alignment, and comparison with an A* baseline across four LLMs. Section 4 discusses limitations and Section 5 concludes. The appendix provides prompting strategies (Appendix A), the full coach vision agent prompt (Appendix B), qualitative flight-plan examples (Appendix C), a per-preference metric breakdown (Appendix E), and a per-difficulty validity analysis (Appendix D).

2. System Architecture

FRAMe's architecture combines an LLM-based planning core with auxiliary modules for solution evaluation and knowledge retrieval. The system takes two main inputs: a flight scenario comprised of an origin, a destination, the flyable airspace, and a set of no-fly zones encoded as polygons in a KML file, and a natural language prompt from the flight operator describing how the route should traverse the environment (for example, "Maximize clearance from hazardous polygons."). KML is a file format that can visualize geographic data in earth browsers such as Google Earth (Google Developers, 2023). The flowchart of the system architecture is depicted in Figure 1. We explain FRAMe components here.

2.1. Planner Agent

The planning process begins by extracting the origin, destination, flyable airspace, and no-fly-zone placemarks from the operator's KML file and converting them into



* The flight operator preference messages are used both in the planner LLM prompt and in RAG-based memory

Figure 1. FRAMe System Architecture: The planner LLM generates a flight plan from a natural language prompt and scenario data. A RAG module provides relevant context to the LLM, such as previous similar planning experiences and their evaluations. The multi-modal LLM coach agent evaluates the planner's proposed route for safety and compliance with the preference. Then, it supplies review that will be combined with the flight operator's feedback and will be stored in a database.

natural-language messages. The system messages are generated according to the chosen prompting strategy, priming the LLM with general instructions. Tabrizian et al. (Tabrizian et al., 2025) discussed different prompting strategies based on CoT and showed that the best results are achieved by using a customized version of CoT without providing any examples (Zero-shot). All of the prompting strategies available in FRAMe are described in the Description of Different Prompting Strategies section in Appendix. After system messages are constructed, they will be combined with user messages (natural-language information of the KML) and the flight operator preferences. This will be the input for the LLM-planner. If RAG is enabled, these messages will be augmented with prior successful flight plans based on the flight setting and operator preferences. With the fully assembled prompt, the LLM-planner produces a candidate flight plan which is a sequence of waypoints forming a flight path that begins at the origin, ends at the destination, avoids all no-fly zones, and adheres as closely as possible to the operator's preference. The planner will also generate a reasoning in natural language for better explainability purposes.

2.2. Multi-modal Coach Agent

The multimodal coach agent is implemented using o4-mini alongside rule-based geometric checks and flight operator's

feedback. The agent first verifies plan validity. A plan is considered valid if: (1) All generated waypoints remain within the flyable airspace, (2) the plan correctly connects the designated origin and destination based on the scenario data, and (3) no segment passes through a no-fly zone.

All criteria are evaluated using geometric tools. Following the validity check, the system renders an image of the full planning problem with the proposed solution overlaid (see Figure 2) and assesses preference alignment, confirming whether the generated route satisfies the operator's directional or positional instructions using o4-mini's vision capabilities (for complete coach vision agent prompt see Appendix B). The vision agent has access to both the rendered solution image and the validity evaluation results. Finally, the flight operator can optionally provide free-form feedback on the proposed plan.

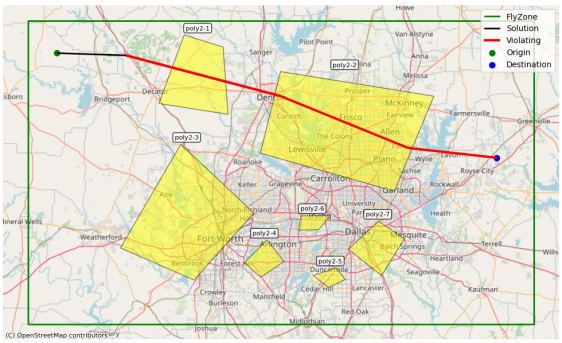


Figure 2. A sample of the generated image of a flight plan for the multi-modal coach agent to review.

2.3. Retrieval Augmented Generation

To augment the prompt with proper context about the current flight planning problem, we implement the RAG module. When enabled, the system queries a vector database to retrieve past flight plans that most closely resemble the current setting and the operator's stated preferences.

The process begins by passing the operator's preferences into an embedding model, which converts the textual input into a numerical embedding vector. This vector is then used by the history retrieval module to identify the most similar prior flight plans based on the cosine distance between the current preference embedding $\mathbf{e_p}$ and each other preference embedding in the database $\forall \mathbf{e_h} \in \mathcal{H}$:

$$\text{CosineDistance}(\mathbf{e_p}, \mathbf{e_h}) = 1 - \frac{\mathbf{e_p} \cdot \mathbf{e_h}}{\|\mathbf{e_p}\| \|\mathbf{e_h}\|},$$

where $\mathcal{H}$ denotes the set of preference embedding vectors corresponding to previously successful flight plans. Since lower cosine distances indicate greater similarity, the system selects records with the K lowest values. Note that the retrieval query restricts the candidate set to prior records with identical scenario geometry, that is, the same flyzone, no-fly polygons, origin, and destination. The cosine similarity over the operator's natural-language preference text is then used only to rank within that geometrically identical set. Geometric relevance is thus guaranteed by exact scenario matching rather than inferred from the embedding distance, and the preference embedding only selects which past plan for the same problem best matches the current intent. The retrieved plans, referred to as the contextual history, represent flight plans most aligned with the current setting and preferences.

The contextual history is then used to augment the input prompt, providing relevant context and guidance for generating the new flight plan. The planner LLM processes this enriched prompt to produce a flight plan, which is presented to the operator for evaluation. If the plan is submitted for storage, the RAG module then collects the relevant data into the vector database, making it available for future retrieval. The complete planning workflow is detailed in Algorithm 1.

Algorithm 1 FRAMe: Multi-modal Coach Agent and RAG

Require: Operator preference P, prompting strategy S, database $\mathcal{DB}$, module flags *RAG* and *Coach*, retrieval count K, scenario C with origin O, destination D, and no-fly zones $Z = \{z_i\}_{i=1}^{n}$
Ensure: Waypoints $W = \{w_j\}_{j=1}^{m}$ and reasoning R

▷ **Step 1: Build prompt**
$sc \leftarrow \text{GetScenario}(C, O, D, Z)$
$M_{\text{sys}} \leftarrow \text{SysMsg}(S)$
$M_{\text{user}} \leftarrow \text{UsrMsg}(P, sc)$
$prompt \leftarrow M_{\text{sys}} + M_{\text{user}}$
$e_p \leftarrow \text{Embed}(P)$
if *RAG* **then**
  $cands \leftarrow \text{Query}(\mathcal{DB}, sc)$
  $\{e_h\}_{h \in \mathcal{H}} \leftarrow \text{EmbedAll}(cands)$
  $dists \leftarrow \{\text{CosDist}(e_h, e_p)\}_{h \in \mathcal{H}}$
  $hist \leftarrow \text{TopK}(K, cands, dists)$
  $ctx \leftarrow \text{GetContext}(hist)$
  $prompt \leftarrow prompt + ctx$
end if
$resp \leftarrow \text{PlannerLLM}(prompt)$
$(W, R) \leftarrow \text{ParsePlan}(resp)$
if *Coach* **then**
  ▷ **Step 2: Validity and alignment checks**
  $val \leftarrow \text{Coach.EvalValidity}(W, sc)$
  $aln \leftarrow \text{Coach.EvalAlignment}(W, sc, \text{SolutionImg})$
  ▷ **Step 3: Operator feedback**
  $fb \leftarrow \text{GetFeedback}(W, R, val, aln, sc)$
  ▷ **Step 4: Update database**
  $rec \leftarrow \{sc, e_p, val, aln, fb, W, R\}$
  $\text{Insert}(\mathcal{DB}, rec)$
end if
return W, R

3. Web Prototype and User Interface

Conducting real flight tests with full-scale eVTOL aircraft is prohibitively expensive at this stage of research. Although

FRAMe is designed and evaluated in the context of eVTOL mission planning, we deploy it as a web application targeting subscale UAV operations as a practical intermediate step. The web interface connects with small UAVs and ground control station software such as Mission Planner (ArduPilot Development Team, 2026), a widely used platform for configuring, controlling, and monitoring UAVs. This is intended to allow validation of the planning pipeline in real flight conditions at a fraction of the cost, while the underlying planning logic and preference-alignment framework remain directly applicable to eVTOL operations. The backend is implemented in Python, integrating the LLM via an API and our custom modules for retrieval and validation. The front-end is a user-friendly interface that guides the operator through scenario setup, preference input, and reviewing the flight plan.

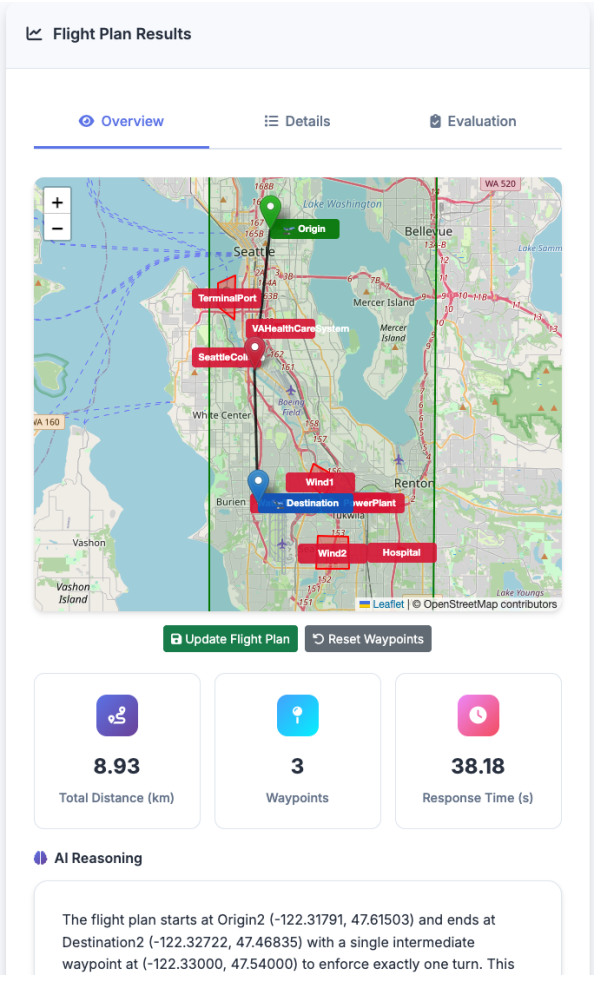


Figure 3. Overview tab in the web user interface: a leaflet map shows the computed route (black) with origin (green), destination (blue), and no-fly zone boundaries (red). AI-reasoning will be provided in this tab.

3.1. Experimental Setup

We evaluate FRAMe on flight scenarios in the Dallas–Fort Worth metropolitan area grouped into three difficulty levels—*Easy*, *Medium*, and *Hard*—that differ in the number and geometry of no-fly zones (2, 4, and 7 polygonal restrictions, respectively). Each difficulty level is paired with every combination of a fixed set of origins and destinations, producing a broad coverage of flight configurations.

Models. We benchmark four state-of-the-art LLMs as the planning module: *OpenAI o3-mini*, *OpenAI o4-mini*, *DeepSeek-R1*, and *OpenAI GPT-5.4*.

Conditions. Each scenario is evaluated under four conditions that reflect progressively richer augmentation. We use the same short names (**A***, **Baseline**, **+RAG**, **+RAG+Coach**) throughout the text, tables, and figures:

1. **A***: a classical A* path planner serving as a geometry-optimal baseline.
2. **Baseline**: the LLM planner alone, with no retrieval augmentation or coach feedback.
3. **+RAG**: the LLM planner augmented with the RAG module, retrieving the $K=2$ most relevant prior plans from the database (no coach).
4. **+RAG+Coach**: the full FRAMe system, combining RAG-based retrieval with the multi-modal coach agent for validity checking and preference verification.

Preferences. To enable objective, quantitative evaluation of preference alignment, we use three measurable preference objectives: (1) *minimize total flight distance*, (2) *minimize the number of waypoints*, and (3) *maximize clearance from hazardous polygons*. For each preference, alignment is assessed by whether the model's output improves the corresponding metric relative to the other preference conditions within the same scenario.

Warmup and ablation protocol. Before ablation, a warmup phase runs each planner over every scenario with a neutral preference prompt, both with and without the coach agent, retaining only the plans that pass the geometric validity checks, to seed the RAG vector database (150 runs). The ablation phase then fixes the database in read-only mode and sweeps each model across the full scenario–preference–condition grid, isolating the marginal contribution of RAG and of the coach.

3.2. Results

Plan validity. Figure 4 reports the validity rate for all four models averaged over difficulty levels, and Table 1 provides the corresponding preference-metric breakdown for valid plans. For every model, +RAG+Coach

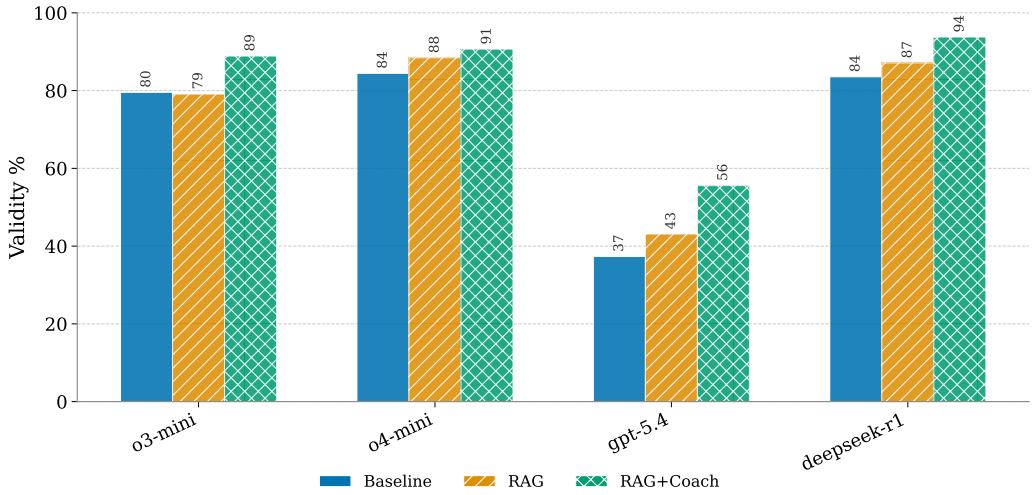


Figure 4. Validity rate (%) by condition for all planners. Conditions are Baseline, +RAG, and +RAG+Coach.

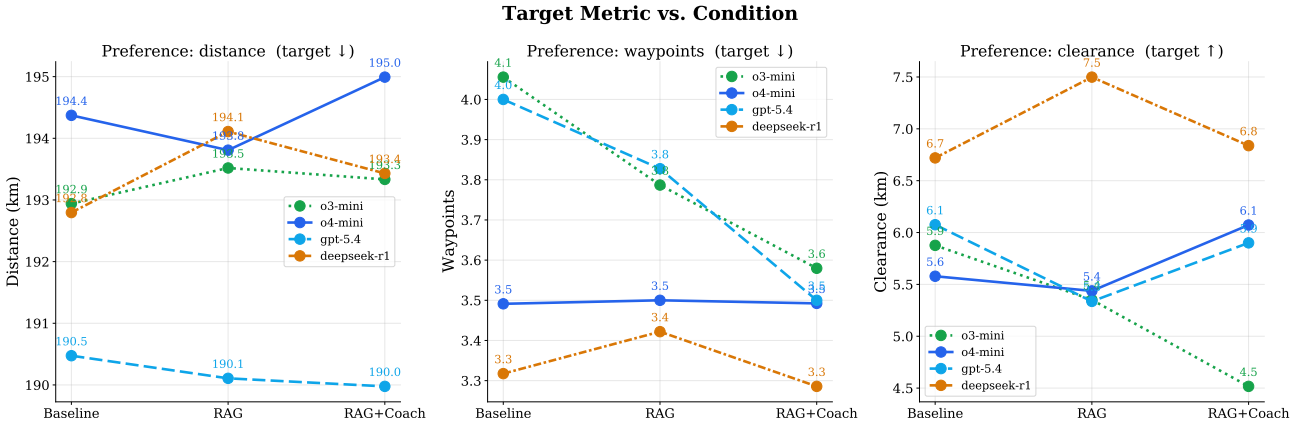


Figure 5. Mean of the preference-relevant metric versus condition for each of the three preferences (valid plans only). Desired direction marked above each panel.

achieves the highest validity. The three reasoning models converge to 88.9–93.8% under the full system (o3-mini: 79.6%→79.1%→88.9%, o4-mini: 84.4%→88.4%→90.7%, DeepSeek-R1: 83.6%→87.1%→93.8%), while GPT-5.4 lags considerably, improving from a markedly lower Baseline of 37.3% to 55.6%—a gap that persists across all conditions and reflects weaker instruction-following on geometrically constrained scenarios. For *o3-mini* in particular, +RAG alone slightly regresses validity (79.6%→79.1%)—retrieval without validation surfaces misleading neighbors—and the coach is what recovers and extends the gain, indicating that the coach review is load-bearing rather than a redundant signal on top of retrieval.

Preference capture. Figure 5 traces how the preference-relevant metric evolves across conditions for valid plans, with reasoning models showing distinct preference-specific patterns. Under the *waypoints* preference, o3-mini is the strongest reasoning-model responder, reducing mean waypoint count from 4.06 (Baseline) to 3.58 (+RAG+Coach, −12%); o4-mini and DeepSeek-R1 start already near their planning floor (3.49 and 3.32, respectively), leaving little room for further reduction. Under the *clearance* preference, o4-mini shows the clearest gain, increasing minimum clearance from 5.58 km to 6.07 km (+9%); DeepSeek-R1 peaks at +RAG (7.50 km, up from 6.72 km at Baseline) but retreats slightly with the coach (6.84 km), suggesting that retrieval alone is the stronger signal for its spatial reasoning. Notably, o3-mini's clearance declines monotonically with augmentation (5.88→5.35→4.52 km in Figure 5; the per-preference matrix in Appendix E reports the same downward trend with rounding to one decimal)—coach-driven revisions appear to prioritize correcting validity violations over maximizing polygon separation. The *distance* preference moves little for all three reasoning models (193–197 km), reflecting that they already operate close to their planning floor at Baseline. GPT-5.4, by contrast, is the only model that mono-

Table 1. Preference-metric results (valid plans only, averaged over all difficulties and preferences). Δ is relative to A*: positive Δ is favorable for Clr. but unfavorable for Dist. (LLMs fly farther than A*) and #WP. Best LLM value per column in **bold**.

Model	Cond.	Dist. (km, Δ)	Clr. (km, Δ)	#WP (Δ)
A*	—	189.9	1.91	18.2
o3-mini	Baseline	195.5 (+5.6)	4.65 (+2.74)	5.17 (−13.0)
	+RAG	194.7 (+4.8)	4.36 (+2.45)	4.65 (−13.5)
	+RAG+Coach	194.5 (+4.6)	3.55 (+1.64)	4.42 (−13.8)
o4-mini	Baseline	195.2 (+5.3)	3.90 (+1.99)	3.94 (−14.3)
	+RAG	196.1 (+6.2)	4.03 (+2.12)	3.81 (−14.4)
	+RAG+Coach	196.6 (+6.7)	4.36 (+2.45)	**3.75 (−14.4)**
GPT-5.4	Baseline	**190.5 (+0.6)**	4.98 (+3.07)	5.36 (−12.8)
	+RAG	191.7 (+1.8)	4.43 (+2.52)	4.68 (−13.5)
	+RAG+Coach	191.2 (+1.3)	4.20 (+2.29)	4.23 (−14.0)
DeepSeek-R1	Baseline	195.7 (+5.8)	4.75 (+2.84)	3.80 (−14.4)
	+RAG	196.6 (+6.7)	**5.25 (+3.34)**	3.83 (−14.4)
	+RAG+Coach	195.9 (+6.0)	5.01 (+3.10)	**3.75 (−14.4)**

tonically improves distance *when the distance preference is active* (190.5→190.0 km in Figure 5); we note this is a per-preference effect—in the all-preferences aggregate of Table 1 GPT-5.4's mean distance rises slightly with augmentation (190.5→191.7→191.2 km), so the gain is conditional on the operator actually requesting distance. GPT-5.4 also shows comparable waypoint reduction (-13%), though its lower overall validity limits the practical impact of these gains.

A* baseline comparison. The classical A* planner achieves 100% validity by construction, but it is preference-blind: its mean clearance of 1.91 km is well below what every LLM produces even in the Baseline condition (3.9–5.0 km), and its waypoint count (18.2) is 3–5$\times$ higher than every LLM configuration because A* tracks the grid discretization rather than the operator's intent. FRAMe (even without RAG) already outperforms A* on the clearance and waypoint preferences, while accepting a small distance overhead over the preference-blind geodesic. Adding +RAG+Coach widens the clearance and waypoint margins and recovers a substantial part of the validity gap for the stronger planners, most clearly for DeepSeek-R1.

4. Limitations

Headroom-bounded preference capture. The distance preference does not move meaningfully under any condition (Table 1) because every model already plans within 6 km of the ≈190 km geodesic at Baseline. The same headroom caveat partially applies to waypoint count for o4-mini and DeepSeek-R1, which sit near a 3.3–3.5 floor at Baseline and therefore have limited room to improve further. Our preference-capture evidence is therefore strongest for the clearance preference (and for waypoint count on planners that start above the floor, e.g. o3-mini and GPT-5.4) and weakest for distance—a consequence of the evaluation geometry rather than evidence that the framework ignores the preference.

Aggregate vs. per-preference effects. Several of our positive findings are clearest when conditioned on the matching preference (Figure 5). When metrics are averaged across all three preferences (Table 1), augmentation can move a non-target metric in the unfavorable direction—for example, aggregate clearance for o3-mini and GPT-5.4 declines under +RAG+Coach as those configurations correctly deprioritize clearance when the operator did not request it. The aggregate column should therefore be read as a side-effect summary, not as the primary preference-capture signal.

LLM-as-judge bias in the coach. The preference-alignment step of the coach is performed by a multimodal LLM (o4-mini) evaluating the planner's output, and prior work has shown that LLM judges can exhibit position, verbosity, and self-preference biases (Zheng et al., 2023). We mitigate this by combining the vision judgment with rule-based geometric checks for validity, but the alignment verdict itself inherits the hidden biases or priors the judge carries. A human-rater study on a subset of plans is left for future work.

Simulation-only evaluation. All experiments use synthetic polygonal no-fly zones; no live flight tests or real weather data are incorporated. As a next step, we plan to validate the approach through real flight tests with a subscale drone.

Constraint coverage. Our current formulation treats hazards as static no-fly polygons and does not model dynamic traffic from other UAVs or crewed aircraft, onboard energy or battery limits, or vehicle dynamics. These factors are safety critical for feasibility and collision avoidance in dense airspace, and incorporating them, for example by adding traffic and energy terms to the scenario and the validity checks, is important future work.

5. Conclusion

We presented FRAMe, an end-to-end flight planner that couples an LLM with a RAG memory of prior plans and a multi-modal coach agent that gates both validity and preference alignment. The framework uses geometric ground truth, so it needs no human annotators, and it is modular: retrieval and the coach can be ablated independently against a fixed warmup-seeded database. Across four state-of-the-art LLMs and three operator preferences, the coach review is load-bearing on top of retrieval, attaining the highest validity for every planner; for the weakest planner, retrieval alone can slightly regress validity, and the coach is what recovers it. Preference capture is real but model-specific: planners with headroom shift the requested metric in the operator-favored direction, while those already near their planning floor have little room to move, and gains can surface under

retrieval alone or only once the coach is added. Finally, FRAMe is preference-aligned where A* is preference-blind, accepting a small distance overhead in exchange for higher clearance and substantially fewer waypoints.

References

ArduPilot Development Team. Mission planner overview. https://ardupilot.org/planner/docs/mission-planner-overview.html, 2026. Accessed: 2026-04-15.

Chen, G., Yu, X., Ling, N., and Zhong, L. TypeFly: Low-Latency Drone Planning With Large Language Models . *IEEE Transactions on Mobile Computing*, 24(09):9068–9079, September 2025. ISSN 1558-0660. doi: 10.1109/TMC.2025.3561282. URL https://doi.ieeecomputersociety.org/10.1109/TMC.2025.3561282.

Chen, Z., Deng, H., Li, Z., Wen, H., Jin, G., Yu, R., and Leng, B. HCRMP: An LLM-Hinted Contextual Reinforcement Learning Framework for Autonomous Driving. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=1BOiVpBtZy.

Dagan, G., Keller, F., and Lascarides, A. Dynamic Planning with a LLM. In *Language Gamification - NeurIPS 2024 Workshop*, 2024. URL https://openreview.net/forum?id=ewx2RFiEYR.

Deng, H., Zhang, H., Ou, J., and Feng, C. Can LLM Be a Good Path Planner Based on Prompt Engineering? Mitigating the Hallucination for Path Planning. In *Advanced Intelligent Computing Technology and Applications: 21st International Conference, ICIC 2025, Ningbo, China, July 26–29, 2025, Proceedings, Part XXIII*, pp. 3–15, Berlin, Heidelberg, 2025. Springer-Verlag. ISBN 978-981-95-0013-0. doi: 10.1007/978-981-95-0014-7_1. URL https://doi.org/10.1007/978-981-95-0014-7_1.

Google Developers. KML Tutorial. https://developers.google.com/kml/documentation/kml_tut, 2023. Last updated November 3, 2023.

Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., Xue, B., Wang, B., Wu, B., Feng, B., Lu, C., Zhao, C., Deng, C., Ruan, C., Dai, D., Chen, D., Ji, D., Li, E., Lin, F., Dai, F., Luo, F., Hao, G., Chen, G., Li, G., Zhang, H., Xu, H., Ding, H., Gao, H., Qu, H., Li, H., Guo, J., Li, J., Chen, J., Yuan, J., Tu, J., Qiu, J., Li, J., Cai, J. L., Ni, J., Liang, J., Chen, J., Dong, K., Hu, K., You, K., Gao, K., Guan, K., Huang, K., Yu, K., Wang, L., Zhang, L., Zhao, L., Wang, L., Zhang, L., Xu, L., Xia, L., Zhang, M., Zhang, M., Tang, M., Zhou, M., Li, M., Wang, M., Li, M., Tian, N., Huang, P., Zhang, P., Wang, Q., Chen, Q., Du, Q., Ge, R., Zhang, R., Pan, R., Wang, R., Chen, R. J., Jin, R. L., Chen, R., Lu, S., Zhou, S., Chen, S., Ye, S., Wang, S., Yu, S., Zhou, S., Pan, S., Li, S. S., Zhou, S., Wu, S., Yun, T., Pei, T., Sun, T., Wang, T., Zeng, W., Liu, W., Liang, W., Gao, W., Yu, W., Zhang, W., Xiao, W. L., An, W., Liu, X., Wang, X., Chen, X., Nie, X., Cheng, X., Liu, X., Xie, X., Liu, X., Yang, X., Li, X., Su, X., Lin, X., Li, X. Q., Jin, X., Shen, X., Chen, X., Sun, X., Wang, X., Song, X., Zhou, X., Wang, X., Shan, X., Li, Y. K., Wang, Y. Q., Wei, Y. X., Zhang, Y., Xu, Y., Li, Y., Zhao, Y., Sun, Y., Wang, Y., Yu, Y., Zhang, Y., Shi, Y., Xiong, Y., He, Y., Piao, Y., Wang, Y., Tan, Y., Ma, Y., Liu, Y., Guo, Y., Ou, Y., Wang, Y., Gong, Y., Zou, Y., He, Y., Xiong, Y., Luo, Y., You, Y., Liu, Y., Zhou, Y., Zhu, Y. X., Huang, Y., Li, Y., Zheng, Y., Zhu, Y., Ma, Y., Tang, Y., Zha, Y., Yan, Y., Ren, Z. Z., Ren, Z., Sha, Z., Fu, Z., Xu, Z., Xie, Z., Zhang, Z., Hao, Z., Ma, Z., Yan, Z., Wu, Z., Gu, Z., Zhu, Z., Liu, Z., Li, Z., Xie, Z., Song, Z., Pan, Z., Huang, Z., Xu, Z., Zhang, Z., and Zhang, Z. DeepSeek-R1 Incentivizes Reasoning in LLMs Through Reinforcement Learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Hirsch, E., Uziel, G., and Anaby-Tavor, A. What's the Plan? Evaluating and Developing Planning-Aware Techniques for Language Models, 2024. URL https://arxiv.org/abs/2402.11489.

ichter, b., Brohan, A., Chebotar, Y., Finn, C., Hausman, K., Herzog, A., Ho, D., Ibarz, J., Irpan, A., Jang, E., Julian, R., Kalashnikov, D., Levine, S., Lu, Y., Parada, C., Rao, K., Sermanet, P., Toshev, A. T., Vanhoucke, V., Xia, F., Xiao, T., Xu, P., Yan, M., Brown, N., Ahn, M., Cortes, O., Sievers, N., Tan, C., Xu, S., Reyes, D., Rettinghouse, J., Quiambao, J., Pastor, P., Luu, L., Lee, K.-H., Kuang, Y., Jesmonth, S., Joshi, N. J., Jeffrey, K., Ruano, R. J., Hsu, J., Gopalakrishnan, K., David, B., Zeng, A., and Fu, C. K. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. In Liu, K., Kulic, D., and Ichnowski, J. (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 287–318. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/ichter23a.html.

Kuffner, J. and LaValle, S. RRT-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Pro-*

ceedings (Cat. No.00CH37065), volume 2, pp. 995–1001 vol.2, 2000. doi: 10.1109/ROBOT.2000.844730.

LaValle, S. M. *Planning Algorithms*. Cambridge university press, 2006.

Liang, K., Zhang, Z., and Fisac, J. F. Introspective Planning: Aligning Robots' Uncertainty with Inherent Task Ambiguity. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=4TlUE0ufiz.

Liu, B., Jiang, Y., Zhang, X., Liu, Q., Zhang, S., Biswas, J., and Stone, P. LLM+P: Empowering Large Language Models with Optimal Planning Proficiency. *arXiv preprint arXiv:2304.11477*, 2023.

Memmel, M., Berg, J., Chen, B., Gupta, A., and Francis, J. STRAP: Robot Sub-Trajectory Retrieval for Augmented Policy Learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4VHiptx7xe.

Meng, S., Wang, Y., Yang, C.-F., Peng, N., and Chang, K.-W. LLM-A*: Large Language Model Enhanced Incremental Heuristic Search on Path Planning. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1087–1102, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.60. URL https://aclanthology.org/2024.findings-emnlp.60/.

OpenAI. Models. OpenAI Developer Platform, 2026. URL https://developers.openai.com/api/docs/models. Accessed: April 24, 2026.

Tabrizian, A., Gupta, P., Taye, A., Jones, J., Thompson, E., Chen, S., Bonin, T., Eberle, D., and Wei, P. Using Large Language Models to Automate Flight Planning Under Wind Hazards. In *2024 AIAA DATC/IEEE 43rd Digital Avionics Systems Conference (DASC)*, pp. 1–8, San Diego, CA, USA, 2024. doi: 10.1109/DASC62030.2024.10749512.

Tabrizian, A., Ghazanfari, M., and Wei, P. Chain-of-Thought Flight Planner: End-to-End LLM Routing Under Wind Hazards. In *AIAA AVIATION FORUM AND ASCEND 2025*, Las Vegas, Nevada, July 2025. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-738-2. doi: 10.2514/6.2025-3711. URL https://arc.aiaa.org/doi/10.2514/6.2025-3711.

Wang, W., Li, Y., Jiao, L., and Yuan, J. GSCE: a Prompt Framework With Enhanced Reasoning for Reliable LLM-Driven Drone Control. In *2025 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 441–448, 2025. doi: 10.1109/ICUAS65942.2025.11007864.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., ichter, b., Xia, F., Chi, E., Le, Q. V., and Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022.

Wu, E. and Mitra, S. Can LLMs plan paths with extra hints from solvers? *arXiv preprint arXiv:2410.05045*, 2024.

Xu, W., Wang, M., Zhou, W., and Li, H. P-RAG: Progressive Retrieval Augmented Generation For Planning on Embodied Everyday Task. In *Proceedings of the 32nd ACM International Conference on Multimedia*, MM '24, pp. 6969–6978, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400706868. doi: 10.1145/3664647.3680661. URL https://doi.org/10.1145/3664647.3680661.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., Zhang, H., Gonzalez, J. E., and Stoica, I. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=uccHPGDlao.

A. Description of Different Prompting Strategies

Here we describe all of the prompting strategies available for FRAMe:

Raw Prompt: This baseline uses minimal prompt engineering. The scenario and request are provided directly to the LLM without any additional guidance on the solution approach. Specifically, the system message is:

Raw Prompt

"You are a flight planner for an eVTOL aircraft. The user will provide you with wind hazard polygon information and request a flight plan from an origin to a destination. You must generate a flight plan as a list of waypoints starting from the origin coordinate and ending at the destination coordinate while avoiding the wind polygons. Always include both the origin and destination points in your response. You can generate as many waypoints as necessary to avoid the polygons. More waypoints may lead to a smoother flight plan. You cannot fly outside of the fly zone."

There is no task breakdown or few-shot example. This represents how the model might perform "out of the box" given a straightforward request.

Zero-Shot Prompt (Basic): In the zero-shot setting, we append the phrase "Think step by step." to the system messages, following (Wei et al., 2022), to prompt a strategic reasoning process. However, we do not specify the individual steps the LLM should follow for planning, nor do we supply any examples in this approach.

Zero-Shot Prompt (Customized): This approach is a modified form of zero-shot prompting that provides more explicit guidance for reasoning. The system message is extended with instructions tailored for chain-of-thought reasoning in flight planning tasks. The following steps are incorporated into the system messages:

Zero-Shot Prompt (Customized)

"You are a flight planner for an eVTOL aircraft. The user will provide you with hazardous polygon information and request a flight plan from an origin to a destination. You must generate a flight plan as a list of waypoints starting from the origin coordinate and ending at the destination coordinate while avoiding the hazardous polygons. Always include both the origin and destination points in your response. You can generate as many waypoints as necessary to avoid the polygons—more waypoints may lead to a smoother flight plan. You cannot fly outside of the flyzone. The best approach to find the optimal solution is as follows: (1) Identify the origin and destination points. (2) Identify the hazardous polygons and the flyzone. (3) IMPORTANT STEP: Generate waypoints that connect the origin to the destination while avoiding hazardous polygons and staying within the flyzone (they should not be on the flyzone's border either). You may generate more waypoints near the hazardous polygons to ensure that the line segments do not intersect with the hazardous polygons. Ensure that the flight plan connecting the waypoints are aligned with the human preference. YOU NEED TO INCLUDE AT LEAST 4 DECIMAL POINTS FOR WAYPOINT COORDINATES. (4) The line segments connecting the waypoints should not have sharp angles (recommended). (5) Ensure that the line segments do not intersect with the hazardous polygons. (6) If any of the line segments intersect with the polygons, modify the corresponding waypoints so the new line segment does not intersect the polygon."

By clarifying the criteria and encouraging intermediate reasoning, this strategy is expected to help the LLM perform better. However, no example flight plan is given, so this remains a zero-shot approach (the model must draw on its internal knowledge and the on-the-fly reasoning).

One-Shot Prompt (Easy Example): This method resembles the zero-shot (customized) approach, but additionally supplies the LLM with a single example problem–solution pair before presenting the new scenario. The example is a simple case (structurally similar but with different coordinates) that illustrates the format of a correct solution.

One-Shot Prompt (Hard Example): This approach is analogous to the previous one, except that the provided example depicts a challenging scenario with multiple hazards, illustrating a more elaborate chain-of-thought process.

B. Coach Vision Agent Prompt

The multi-modal coach agent receives two inputs simultaneously: (i) a structured text prompt, shown below, in which {geometric_summary} is filled at runtime with the output of the rule-based geometric checks (validity flag, violated polygons, waypoints outside the flyzone, and origin/destination match), and {human_preference} with the operator's stated preference; and (ii) the rendered flight plan image depicted in Figure 2. The agent first handles invalid plans (Step 0), then—only if the plan is geometrically valid—judges preference alignment from the image (Step 1), returning a structured verdict (aligned, evaluation, reasoning).

Coach Vision Agent Prompt

You are a senior flight-operations reviewer evaluating an eVTOL flight plan that was produced by an automated planner.

SCOPE
Validity checks (polygon intersection, flyzone containment, origin/destination endpoints) have already been performed by geometric tools upstream. DO NOT re-evaluate validity. Your ONLY job is to judge how well the flight plan aligns with the flight operator's preference, or—if no preference is given—how optimal the plan looks.

LEGEND (what you will see in the image)

- Green rectangle: the flyzone.
- Yellow polygons (labeled poly1-1, poly1-2, . . .): hazardous zones.
- Green dot labeled "Origin": start of the route.
- Blue dot labeled "Destination": end of the route.
- Black line segments: path segments not intersecting any hazardous polygon.
- Red line segments: path segments intersecting a hazardous polygon. (Informational only—ignore for this review.)

GEOMETRIC EVALUATION (ground truth from upstream tools—trust this over the image)
{geometric_summary}

FLIGHT OPERATOR PREFERENCE
{human_preference}

EVALUATION TASK
STEP 0 — Check validity first (from the GEOMETRIC EVALUATION block above):

- If "Valid overall" is False, the path is INVALID. Do NOT judge preference alignment or optimality. Return: aligned = False, evaluation = ''Path is invalid---alignment not evaluated.'', reasoning = a short summary of which geometric rule was broken. Stop here.

STEP 1 — Only if the path is valid, evaluate alignment:

- If a meaningful preference is provided: judge how well the path follows that preference. Be specific about which part of the route satisfies or violates it. Set aligned = True only if the path clearly follows the preference.
- If no meaningful preference is provided: judge the path on optimality only—is it close to the shortest reasonable route? Are there unnecessary detours, zig-zags, or sharp turns? Set aligned = True if the path is near-optimal, False otherwise.

Use the geometric evaluation above as hard facts. For the alignment judgment, focus on the VISUAL pattern: shape of the detour, which side of polygons the path goes around, spacing of waypoints, distance from hazards. Cite concrete visual evidence. Do not speculate about data you cannot see in the image.

OUTPUT
Return a structured object with: aligned (bool), evaluation (one- or two-sentence verdict).

C. Extended Qualitative Flight-Plan Examples

The examples below show the real inputs and outputs of two valid and one invalid FRAMe runs with the *o4-mini* planner. Within each *FRAMe Input* box, three sections are color-coded: **SYS** (the fixed system message, shared across all runs), **USER** (the per-scenario user message built by prompt_generator), and **OPERATOR** (the operator preference, appended to the user message). The *FRAMe Response* box shows the waypoints and natural-language explanation returned by the planner. Finally, coach review is shown for the generated flight plan. Polygon labels such as poly7-1 and poly1-3 use the scenario identifier as a prefix; the leading number is a scenario ID, not the polygon count.

C.1. Easy scenario, clearance preference

FRAMe Input (SYS + USER + OPERATOR)

SYS: You are a flight planner for an eVTOL aircraft. The user will provide you with hazardous polygon information and request a flight plan from an origin to a destination. You must generate a flight plan as a list of waypoints starting from the origin coordinate and ending at the destination coordinate while avoiding the hazardous polygons. Always include both the origin and destination points in your response. You can generate as many waypoints as necessary to avoid the polygons—more waypoints may lead to a smoother flight plan. You cannot fly outside of the flyzone. The best approach to find the optimal solution is as follows: (1) Identify the origin and destination points. (2) Identify the hazardous polygons and the flyzone. (3) IMPORTANT STEP: Generate waypoints that connect the origin to the destination while avoiding hazardous polygons and staying within the flyzone (they should not be on the flyzone's border either). You may generate more waypoints near the hazardous polygons to ensure that the line segments do not intersect with the hazardous polygons. Ensure that the flight plan connecting the waypoints are aligned with the human preference. YOU NEED TO INCLUDE AT LEAST 4 DECIMAL POINTS FOR WAYPOINT COORDINATES. (4) The line segments connecting the waypoints should not have sharp angles (recommended). (5) Ensure that the line segments do not intersect with the hazardous polygons. (6) If any of the line segments intersect with the polygons, modify the corresponding waypoints so the new line segment does not intersect the polygon.

USER: Now you have to generate a flight plan avoiding the hazardous polygons for the following problem. Coordinates for 'FlyZone': $(-98.082, 33.465)$, $(-98.082, 32.473)$, $(-96.110, 32.473)$, $(-96.110, 33.465)$. Coordinates for 'poly7-1': $(-97.319, 33.093)$, $(-96.749, 33.028)$, $(-96.667, 33.248)$, $(-97.164, 33.304)$. Coordinates for 'poly7-2': $(-97.133, 32.631)$, $(-96.754, 32.587)$, $(-96.679, 32.736)$, $(-96.964, 32.852)$. Coordinates for 'Origin2': $(-97.966, 33.149)$. Coordinates for 'Destination4': $(-96.255, 32.826)$.

OPERATOR: "Maximize clearance from hazardous polygons."

[F] FRAMe Response

Waypoints: $(33.1490, -97.9662) \rightarrow (33.4500, -97.9662) \rightarrow (33.4500, -96.2548) \rightarrow (32.8261, -96.2548)$.

Reasoning: "To maximize clearance from the hazardous polygons, the flight plan goes north to the upper flyzone boundary ($33.45°$N), then east to the destination's longitude, then south to the destination. This path stays far from all polygons."

[F] Coach Review

Valid: True **Waypoints outside flyzone:** {} **Violated polygons:** []

Coach Verdict: [ALIGNED]

Evaluation: Strong alignment with the 'maximize clearance' preference. The route climbs to the northern border, traverses the top edge, and descends at the far east—staying well away from both hazardous polygons.

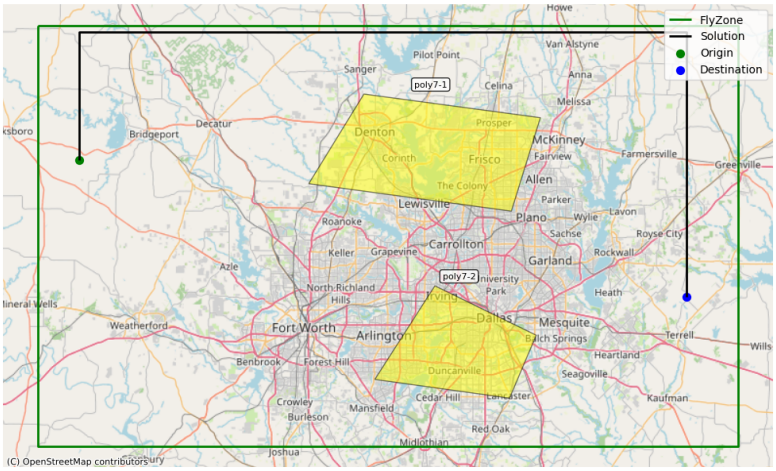


Figure 6. Easy scenario, clearance preference. The generated image of the flight plan.

C.2. Medium scenario, minimize distance preference

FRAMe Input (SYS + USER + OPERATOR)

SYS: You are a flight planner for an eVTOL aircraft. The user will provide you with hazardous polygon information and request a flight plan from an origin to a destination. You must generate a flight plan as a list of waypoints starting from the origin coordinate and ending at the destination coordinate while avoiding the hazardous polygons. Always include both the origin and destination points in your response. You can generate as many waypoints as necessary to avoid the polygons—more waypoints may lead to a smoother flight plan. You cannot fly outside of the flyzone. The best approach to find the optimal solution is as follows: (1) Identify the origin and destination points. (2) Identify the hazardous polygons and the flyzone. (3) IMPORTANT STEP: Generate waypoints that connect the origin to the destination while avoiding hazardous polygons and staying within the flyzone (they should not be on the flyzone's border either). You may generate more waypoints near the hazardous polygons to ensure that the line segments do not intersect with the hazardous polygons. Ensure that the flight plan connecting the waypoints are aligned with the human preference. YOU NEED TO INCLUDE AT LEAST 4 DECIMAL POINTS FOR WAYPOINT COORDINATES. (4) The line segments connecting the waypoints should not have sharp angles (recommended). (5) Ensure that the line segments do not intersect with the hazardous polygons. (6) If any of the line segments intersect with the polygons, modify the corresponding waypoints so the new line segment does not intersect the polygon.

USER: Now you have to generate a flight plan avoiding the hazardous polygons for the following problem. Coordinates for 'FlyZone': $(-98.082, 33.465)$, $(-98.082, 32.473)$, $(-96.110, 32.473)$, $(-96.110, 33.465)$. Coordinates for 'poly4-1': $(-97.497, 33.306)$, $(-97.254, 33.140)$, $(-97.039, 33.329)$, $(-97.181, 33.436)$. Coordinates for 'poly4-2': $(-97.094, 32.993)$, $(-96.871, 32.867)$, $(-96.601, 33.051)$, $(-96.749, 33.185)$. Coordinates for 'poly4-3': $(-97.514, 32.640)$, $(-97.263, 32.632)$, $(-97.433, 33.063)$, $(-97.720, 32.985)$. Coordinates for 'poly4-4': $(-97.022, 32.547)$, $(-96.739, 32.596)$, $(-96.769, 32.706)$, $(-97.021, 32.635)$. Coordinates for 'Origin3': $(-97.966, 32.893)$. Coordinates for 'Destination4': $(-96.255, 32.826)$. Human preference: Minimize total flight distance.

OPERATOR: "Minimize total flight distance."

[F] FRAMe Response

Waypoints: $(32.8931, -97.9655) \rightarrow (32.6300, -97.5137) \rightarrow (32.8261, -96.2548)$.

Reasoning: "The direct path from origin to destination intersects poly4-3. To avoid it, we add a waypoint just south of poly4-3's bottom-left vertex, ensuring the flight path stays clear of all hazardous polygons and within the flyzone."

[F] Coach Review

Valid: False **Waypoints outside flyzone:** {} **Violated polygons:** [poly4-3]

Coach Verdict: [NOT ALIGNED]

Evaluation: Path is invalid—alignment not evaluated.

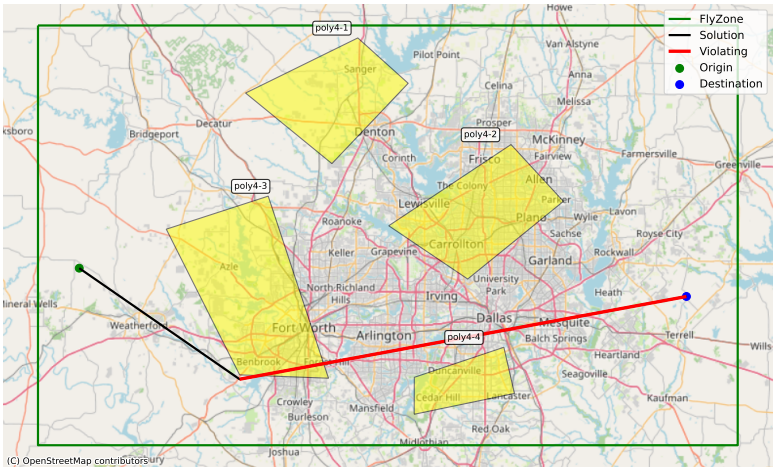


Figure 7. Medium scenario, minimize distance preference. The generated image of the flight plan.

C.3. Hard scenario, minimize waypoints preference

FRAMe Input (SYS + USER + OPERATOR)

SYS: You are a flight planner for an eVTOL aircraft. The user will provide you with hazardous polygon information and request a flight plan from an origin to a destination. You must generate a flight plan as a list of waypoints starting from the origin coordinate and ending at the destination coordinate while avoiding the hazardous polygons. Always include both the origin and destination points in your response. You can generate as many waypoints as necessary to avoid the polygons—more waypoints may lead to a smoother flight plan. You cannot fly outside of the flyzone. The best approach to find the optimal solution is as follows: (1) Identify the origin and destination points. (2) Identify the hazardous polygons and the flyzone. (3) IMPORTANT STEP: Generate waypoints that connect the origin to the destination while avoiding hazardous polygons and staying within the flyzone (they should not be on the flyzone's border either). You may generate more waypoints near the hazardous polygons to ensure that the line segments do not intersect with the hazardous polygons. Ensure that the flight plan connecting the waypoints are aligned with the human preference. YOU NEED TO INCLUDE AT LEAST 4 DECIMAL POINTS FOR WAYPOINT COORDINATES. (4) The line segments connecting the waypoints should not have sharp angles (recommended). (5) Ensure that the line segments do not intersect with the hazardous polygons. (6) If any of the line segments intersect with the polygons, modify the corresponding waypoints so the new line segment does not intersect the polygon.

USER: Now you have to generate a flight plan avoiding the hazardous polygons for the following problem. Coordinates for 'FlyZone': $(-98.082, 33.465)$, $(-98.082, 32.473)$, $(-96.110, 32.473)$, $(-96.110, 33.465)$. Coordinates for 'poly1-1': $(-97.584, 33.194)$, $(-97.241, 33.081)$, $(-97.104, 33.375)$, $(-97.263, 33.425)$. Coordinates for 'poly1-2': $(-97.095, 33.035)$, $(-96.773, 32.814)$, $(-96.434, 33.021)$, $(-96.586, 33.346)$. Coordinates for 'poly1-3': $(-97.643, 32.826)$, $(-97.510, 32.718)$, $(-97.248, 32.905)$, $(-97.416, 33.049)$. Coordinates for 'poly1-4': $(-97.309, 32.612)$, $(-97.208, 32.625)$, $(-97.173, 32.704)$, $(-97.265, 32.694)$. Coordinates for 'poly1-5': $(-97.030, 32.644)$, $(-96.897, 32.625)$, $(-96.898, 32.747)$, $(-97.051, 32.748)$. Coordinates for 'poly1-6': $(-96.559, 32.635)$, $(-96.507, 32.686)$, $(-96.571, 32.768)$, $(-96.629, 32.691)$. Coordinates for 'poly1-7': $(-96.765, 33.428)$, $(-96.954, 33.415)$, $(-96.947, 33.262)$, $(-96.779, 33.324)$. Coordinates for 'Origin1': $(-97.970, 33.359)$. Coordinates for 'Destination1': $(-96.256, 33.415)$. Human preference: Minimize the number of waypoints—keep the path as simple as possible. Previous solutions with coach reviews for similar problems. Example 1: Human preference: Propose the best flight plan from origin to destination avoiding hazardous polygons. Solution waypoints: $(-97.970, 33.359) \rightarrow (-97.200, 33.460) \rightarrow (-96.256, 33.415)$. Valid: True. Waypoints outside flyzone: $\{\}$. Violated polygons: [].

OPERATOR: "Minimize the number of waypoints—keep the path as simple as possible."

[F] FRAMe Response

Waypoints: $(33.3594, -97.9705) \rightarrow (33.4654, -97.2000) \rightarrow (33.4155, -96.2559)$.

Reasoning: "The flight plan starts at the origin, proceeds to an intermediate waypoint set at a high latitude ($33.4654°$N) to safely avoid all hazardous polygons, particularly poly1-1 and poly1-7, and then continues to the destination. This path minimizes the number of waypoints as per human preference while ensuring the aircraft stays within the flyzone and avoids polygon intersections by flying above the obstacles."

[F] Coach Review

Valid: True **Waypoints outside flyzone:** $\{\}$ **Violated polygons:** []

Coach Verdict: [ALIGNED]

Evaluation: Aligned with the preference: the route uses only one intermediate waypoint, keeping the path to two long segments while avoiding hazards.

D. Validity by Difficulty Level

Figure 9 breaks down the full system (+RAG+Coach) by difficulty level and reveals that the relationship between geometric complexity and validity is model-specific rather than uniform. DeepSeek-R1 achieves the highest validity on Easy and Medium scenarios (98.7% and 97.3%, respectively) but drops to 85.3% on Hard, suggesting that its strong internal reasoning

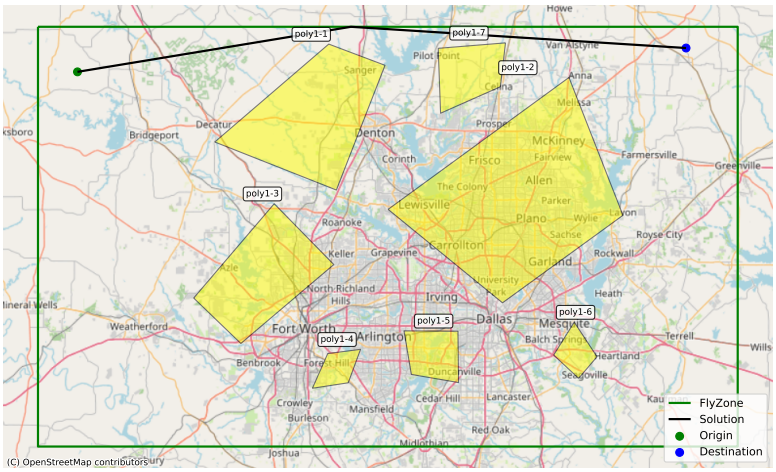


Figure 8. Hard scenario, minimize waypoints preference. The generated image of the flight plan.

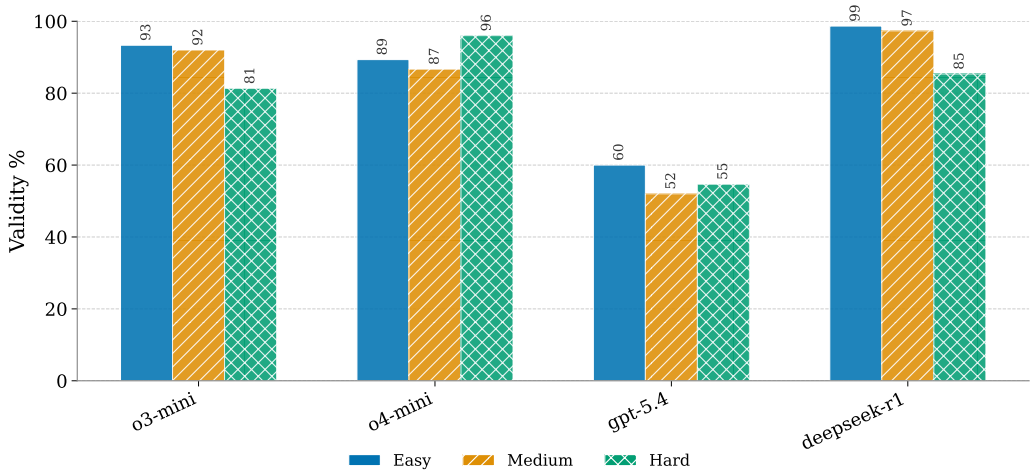


Figure 9. Plan validity (%) of the full FRAMe system (+RAG+Coach) broken down by difficulty level (Easy, Medium, Hard) for each planner. The relationship between difficulty and validity is model-specific: DeepSeek-R1 peaks on Easy/Medium (98.7%/97.3%) while o4-mini peaks on Hard (96.0%), and GPT-5.4 remains consistently the weakest planner across all difficulty levels (52–60%).

becomes less reliable when navigating seven overlapping no-fly zones simultaneously. o4-mini exhibits the opposite pattern: its Hard validity (96.0%) exceeds its Easy (89.3%) and Medium (86.7%) rates, indicating that richer geometric structure helps anchor this model's planning decisions rather than overwhelm them. o3-mini shows a monotone decline from Easy (93.3%) through Medium (92.0%) to Hard (81.3%), consistent with a planner that handles moderate complexity well but degrades under the densest airspace configurations. GPT-5.4 remains consistently the weakest planner across all difficulty levels (52–60%), with no clear sensitivity to difficulty; the coach partially compensates for its lower intrinsic planning capability but cannot fully bridge the gap to the other three models.

E. Per-Preference Capture Matrix

Figures 10 and 11 show the full per-preference breakdown of all three metrics (distance, waypoints, clearance) across the three conditions (Baseline, +RAG, +RAG+Coach) for each model. Each cell reports the mean value over valid plans only. Reading across a row shows how a given metric changes as augmentation increases; reading down a column shows how different preferences affect the same metric under the same condition. The diagonal entries (where the preference matches the metric, e.g. distance metric under the distance preference) are the primary preference-capture signal: a diagonal cell that moves in the desired direction (lower distance or waypoints, higher clearance) relative to its Baseline indicates that the corresponding condition captures the active preference for that model. Off-diagonal entries are informative as side-effect

probes—e.g. asking for clearance can lengthen distance, or asking for fewer waypoints can reduce clearance—and should be read as costs incurred to satisfy a different active preference rather than as failures of preference capture.

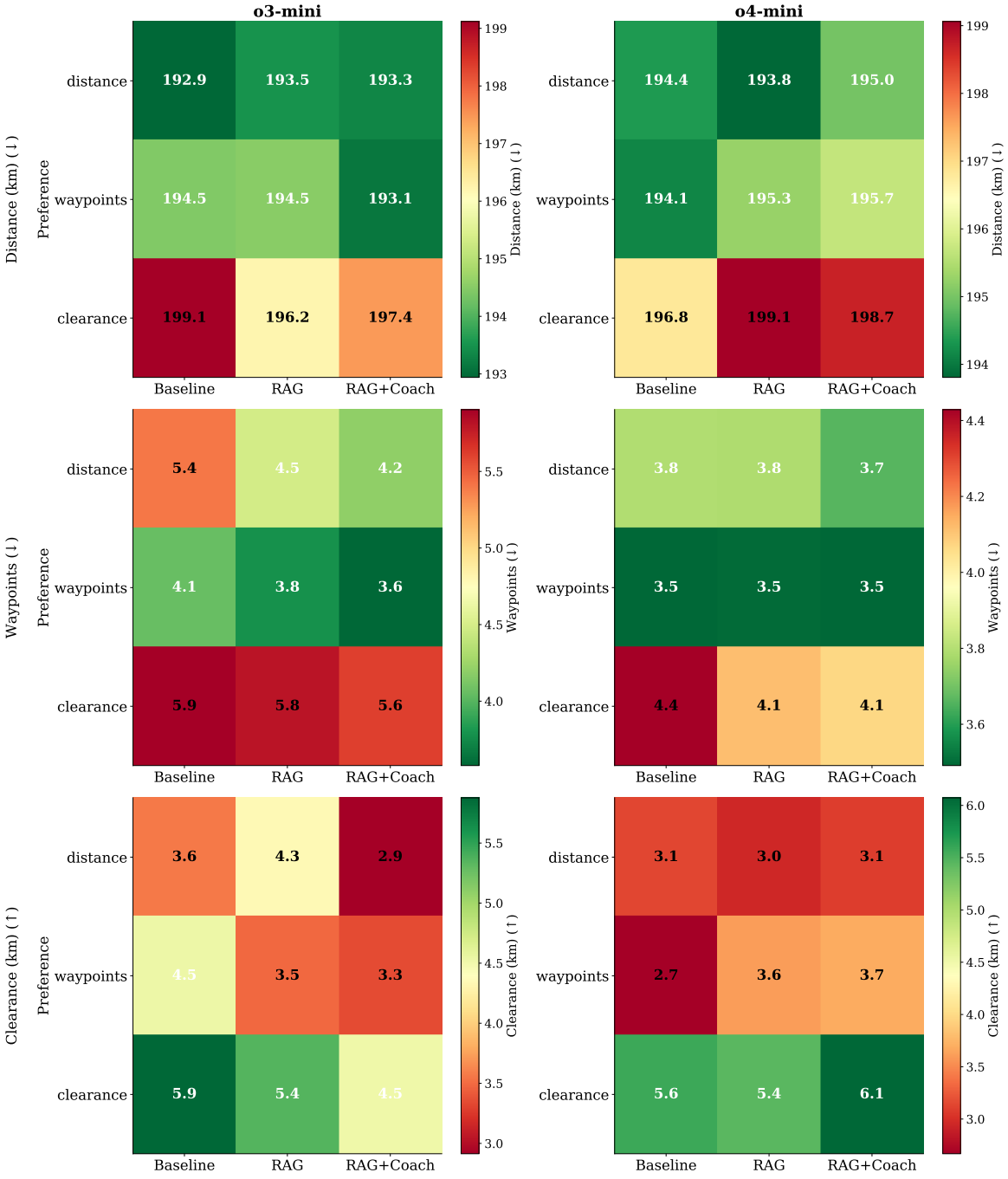


Figure 10. Mean metric values by preference and condition for each model. Each cell reports the average over valid plans (o3-mini and o4-mini).

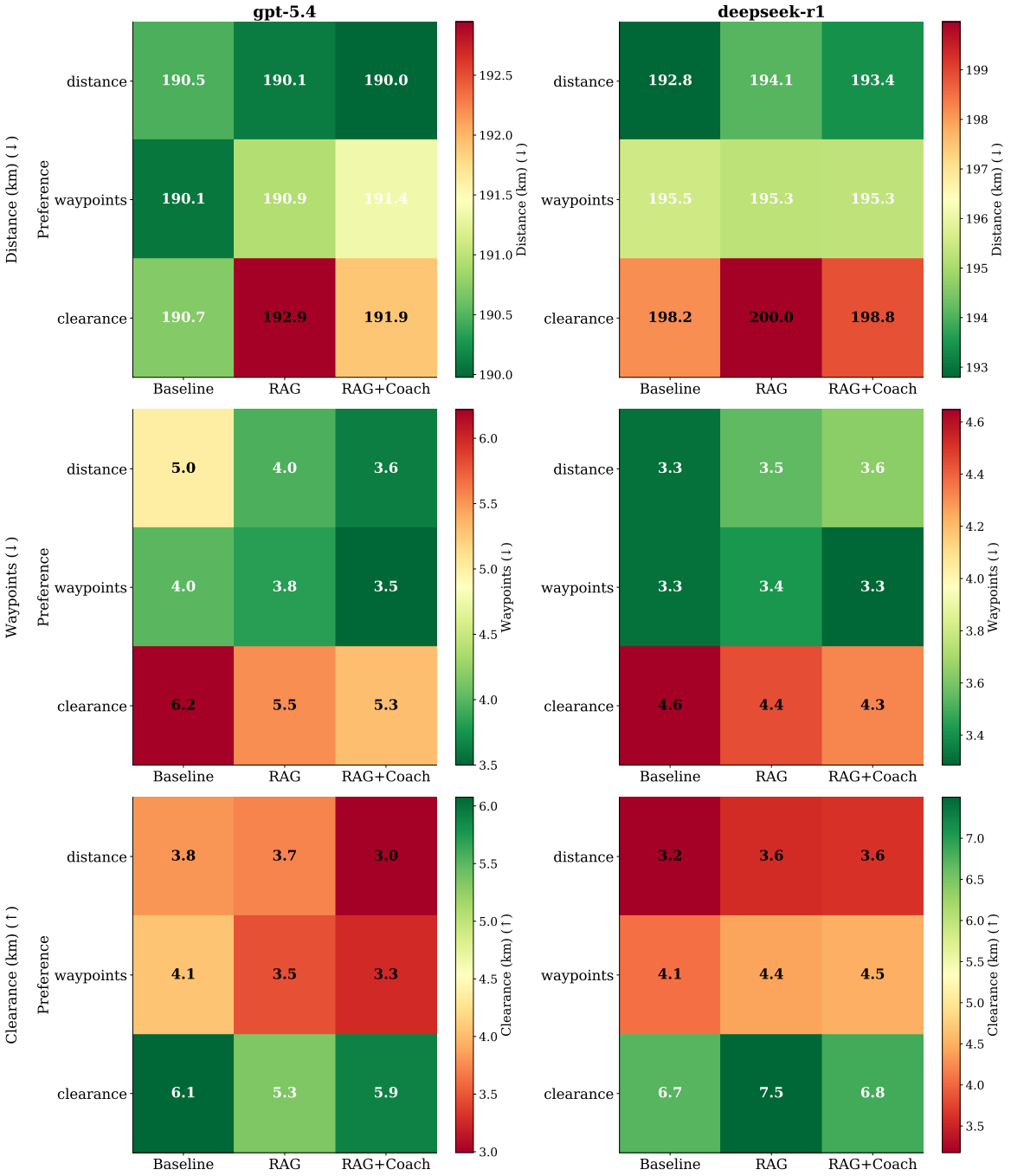


Figure 11. Mean metric values by preference and condition for each model. Each cell reports the average over valid plans(gpt-5.4 and deepseek-r1).