VLM-RRT: Vision Language Model Guided RRT Search for Autonomous UAV Navigation

Jianlin Ye, Savvas Papaioannou and Panayiotis Kolios

***Abstract*— Path planning is a fundamental capability of autonomous Unmanned Aerial Vehicles (UAVs), enabling them to efficiently navigate toward a target region or explore complex environments while avoiding obstacles. Traditional path-planning methods, such as Rapidly-exploring Random Trees (RRT), have proven effective but often encounter significant challenges. These include high search space complexity, sub-optimal path quality, and slow convergence, issues that are particularly problematic in high-stakes applications like disaster response, where rapid and efficient planning is critical. To address these limitations and enhance path-planning efficiency, we propose Vision Language Model RRT (VLM-RRT), a hybrid approach that integrates the pattern recognition capabilities of Vision Language Models (VLMs) with the path-planning strengths of RRT. By leveraging VLMs to provide initial directional guidance based on environmental snapshots, our method biases sampling toward regions more likely to contain feasible paths, significantly improving sampling efficiency and path quality. Extensive quantitative and qualitative experiments with various state-of-the-art VLMs demonstrate the effectiveness of this proposed approach.**

I. Introduction

As Unmanned Aerial Vehicles (UAVs) operate in increasingly dynamic and complex environments, the demand for reliable navigation [1], including efficient and adaptive path-planning strategies [2], has grown significantly. Path planning, a critical component of autonomous UAV navigation, determines the optimal path from a starting point to a target region while avoiding obstacles, often optimizing specific mission objectives [3]–[6]. This process is central to applications such as emergency response [7]–[11], surveillance [12]–[16], and automated inspection [17]–[22].

Existing sampling-based path-planning algorithms, such as Rapidly-exploring Random Trees (RRT) [23], [24], offer significant advantages, including their ability to handle high-dimensional spaces and their probabilistic completeness, meaning they will eventually find a solution if one exists. However, these methods require careful fine-tuning and often fail to converge to optimal solutions, particularly in complex or cluttered environments. This limitation is critical, as it reduces their reliability, making them less suitable for high-stakes applications like search-and-rescue and disaster response missions, where rapid and dependable solutions are paramount. Recent hybrid approaches combining sampling-based methods with machine learning techniques have shown promise in addressing these shortcomings, offering improved computational efficiency and path quality [25]–[27].

The authors are with the KIOS Research and Innovation Centre of Excellence (KIOS CoE), and the Department of Computer Science, University of Cyprus, Nicosia, 1678, Cyprus. {ye.jianlin, papaioannou.savvas, pkolios}@ucy.ac.cy

In this direction, this work integrates multimodal large language models (LLMs) with RRT-based path planning to tackle these challenges. The proposed framework, Vision Language Model RRT (VLM-RRT), combines the strengths of sampling-based path planning through RRT with the pattern-matching capabilities and emergent reasoning of LLMs, thereby enhancing autonomous UAV navigation by enabling efficient and robust path planning. Specifically, the proposed approach incorporates a VLM module into the path-planning process to analyze environmental snapshots and dynamically guide the planner to prioritize and sample from specific regions. This reduces redundant exploration and accelerates convergence by biasing the sampling process toward regions more likely to contain feasible and efficient paths, thus improving convergence rates and path quality. The contributions of this work can be summarized as follows:

- We propose VLM-RRT, a novel framework that augments traditional RRT-based path planning with the advanced reasoning capabilities of Vision Language Models (VLMs). By leveraging a VLM module to analyze environmental snapshots, VLM-RRT enhances navigation decisions by dynamically guiding the sampling process, effectively biasing it toward regions with a higher likelihood of containing optimal paths.
- Extensive qualitative and quantitative experimental evaluations using OpenAI's GPT-4o and Meta's Llama 3.2 multimodal LLMs demonstrate the effectiveness of the proposed approach in terms of sampling efficiency and path quality compared to the standalone RRT approach.

The remainder of this paper is organized as follows. Section II provides background and discusses related work, Section III formulates the problem, and Section IV details the proposed approach. Finally, Section V evaluates the proposed approach, and Section VI concludes the paper.

II. Related Work

The Rapidly-exploring Random Tree (RRT) algorithm [23] is a sampling-based method for path planning, designed to efficiently explore high-dimensional configuration spaces. It incrementally builds a tree by randomly sampling points in the space and connecting them to the nearest point in the existing tree, ensuring rapid exploration. RRT is particularly effective for finding feasible paths in complex, obstacle-filled environments, making it a widely used approach in robotics and autonomous navigation.

Significant algorithmic refinements have enhanced RRT's capabilities over the years. For instance, the RRT* algorithm [28] introduced asymptotic optimality through node-rewiring

mechanisms by minimizing a predefined cost function (e.g., path length), while Informed RRT* [29] developed advanced sampling strategies to guide exploration near optimal solution regions. On the other hand, RRT-Connect [24] grows two trees bidirectionally from both the start and goal configurations. This approach, combined with a greedy heuristic, allows faster exploration of the configuration space and quicker convergence to a solution compared to the standard RRT.

To tackle issues related to random sampling and low path efficiency in RRT, an improved approach incorporating the Artificial Potential Field (APF) method was recently proposed in [30]. This method introduces a probability value during the expansion step of the random tree in the basic RRT algorithm, enhancing convergence speed toward the target node. Similarly, the authors in [31] proposed an RRT variant that utilizes adjustable probability and sampling area strategies to quickly find a feasible path. This planner is then combined with an optimizer that uses the Dijkstra algorithm to prune and improve the initial path. More recently, the work in [32] combined RRT with model predictive control (MPC) utilizing control barrier functions (CBF) to enforce safety-critical constraints. Additionally, recent advancements in learning-based RRT methods have shown promising results in improving path-planning efficiency. The Neural Informed RRT* approach [33] utilizes a neural network to learn the topology of the free space and infer states close to the optimal path, thereby guiding the search toward more promising regions, whereas Neural RRT* [34] employs convolutional neural networks to predict a probabilistic heatmap of states for guiding exploration.

Finally, the emergence of Large Language Models (LLMs) [35], [36] has revolutionized artificial intelligence, enabling advanced reasoning and knowledge-driven applications in autonomous navigation. Recent research has explored the potential of LLMs to enhance navigation tasks by combining their reasoning capabilities with domain-specific methods. For instance, the work in [37] demonstrated that LLMs can perform high-level planning tasks for navigation, including identifying landmarks from observed scenes, tracking navigation progress, and correcting course. More closely related to our work is the approach in [38], where the authors integrated LLMs with the A* path-planning algorithm [39], achieving enhanced pathfinding efficiency in terms of time and space complexity.

In summary, RRT-based path-planning approaches are extensively utilized for their ability to explore state spaces efficiently and effectively. However, they often require meticulous parameter tuning and tend to exhibit slow convergence. While these methods can find optimal solutions, they frequently incur significant computational overhead, with high memory and time demands, particularly when searching for the best path. This limitation is especially critical in applications such as autonomous vehicles, where rapid identification of efficient paths is essential due to constraints like limited power or fuel resources. Motivated by these challenges, this work proposes a novel path-planning framework that integrates the reasoning capabilities of multimodal large language models with RRT-based path search to enhance the efficiency of path generation.

III. Preliminaries

This study addresses a UAV path-planning challenge inspired by real-world wildfire disaster response scenarios. The objective is to autonomously navigate a UAV through fire-affected forested regions to locate survivors while ensuring safe traversal by avoiding hazardous fire fronts. The UAV must reach a predefined goal region while dynamically adapting its trajectory to evolving environmental conditions. To support this task, we assume the presence of a disaster early-warning system (EWS) equipped with multimodal sensing capabilities, including satellite imagery and meteorological data. This system provides real-time updates on the locations of both fire fronts and survivors, enabling the UAV to maintain up-to-date situational awareness. Leveraging this information, the UAV must compute an optimal trajectory that balances mission success with environmental constraints, ensuring efficient and safe navigation through the disaster zone.

A. UAV Dynamical Model

Without loss of generality, we assume that the dynamical behavior $\mathscr{B}$ of a UAV agent can be described by a linear time-invariant (LTI) system [40] of the following form:

$$\mathscr{B}(A,B,C,D) := \begin{cases} x(t+1) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t), \end{cases} \tag{1}$$

where $\mathscr{B}(A,B,C,D)$ is the input/output/state representation of the system, with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, and $D \in \mathbb{R}^{p\times m}$ known. The state, control input, and output of the system at time-step $t \in \mathbb{N}$ are given respectively by $x(t) \in \mathcal{X} \subset \mathbb{R}^n$, $u(t) \in \mathcal{U} \subset \mathbb{R}^m$, and $y(t) \in \mathcal{Y} \subset \mathbb{R}^p$. For conciseness, we assume that $x(t) \in \mathbb{R}^6$ represents the 3D position and linear velocity of the UAV in Cartesian coordinates. The control input force is denoted by $u(t) \in \mathbb{R}^3$, and the UAV position by $y(t) \in \mathbb{R}^3$.

B. Environment Model

The UAV operates within a bounded three-dimensional environment $\mathcal{E} \subset \mathbb{R}^3$, which consists of (a) a designated goal region $\mathcal{G}$ that the UAV must reach during its mission and (b) a set of fire fronts $\mathcal{O}$ that must be avoided. The mission begins at the UAV's home depot $\mathcal{S}$, from which it departs at the start of the search and concludes upon reaching $\mathcal{G}$. All relevant environmental information, including the locations of $\mathcal{S}$, $\mathcal{G}$, and fire fronts $o \in \mathcal{O}$, is assumed to be fully known and provided to the UAV at the outset of the mission. In this formulation, $\mathcal{S}$, $\mathcal{G}$, and the set of fire fronts $\mathcal{O}$ are represented as rectangular cuboids of varying dimensions.

C. Problem Formulation

The autonomous UAV navigation problem, as previously discussed, can be formulated as a finite-horizon optimal control problem, as presented in Eq. (2). The objective is to determine the optimal UAV control inputs $u(t)$, for $t \in \{0, \ldots, T-1\}$, over a suitably chosen planning horizon of T time steps. These control inputs guide the UAV toward the goal region $\mathcal{G}$ by tracking a reference path $\mathcal{P}$ while adhering to the system's dynamics and constraints, as shown.

$$
\begin{aligned}
\min_{u,y} \quad & \sum_{t=0}^{T-1} \left(\|y(t) - \mathcal{P}(t)\|_Q^2 + \|u(t)\|_R^2 \right) \\
\text{s.t.} \quad & x(t+1) = Ax(t) + Bu(t), \quad \forall t \in \{0, \ldots, T-1\}, \\
& y(t) = Cx(t) + Du(t), \quad \forall t \in \{0, \ldots, T-1\}, \\
& x(0) = x_{init}, \\
& x(t) \in \mathcal{X}, \quad \forall t \in \{0, \ldots, T-1\}, \\
& u(t) \in \mathcal{U}, \quad \forall t \in \{0, \ldots, T-1\}, \\
& y(t) \in \mathcal{Y}, \quad \forall k \in \{0, \ldots, N-1\}.
\end{aligned} \tag{2}
$$

In the formulation above, x_{init} represents the agent's initial state, positioning the agent within its home depot $\mathcal{S}$. The objective is to track the reference path $\mathcal{P}(t)$, $t \in \{0, \ldots, T-1\}$ (which describes a feasible path to the goal region), over the time horizon, subject to the UAV's dynamical constraints and operational behavior. The norm $\|u(t)\|_R^2$ denotes the quadratic form $u(t)^\top R u(t)$ (likewise for $\|\cdot\|_Q^2$), where $R \in \mathbb{R}^{m \times m}$ is the control cost matrix and $Q \in \mathbb{R}^{p \times p}$ is the output cost matrix. In essence, the objective function minimizes the weighted sum of the tracking error (i.e., deviation from the reference path) and control effort, with the weighting scheme captured in the matrices Q and R, respectively. The reference path $\mathcal{P}$ provides a collision-free path from $\mathcal{S}$ to the goal region $\mathcal{G}$. Once $\mathcal{P}$ is known, the solution to the optimization problem in Eq. (2) can be obtained using numerical optimization, such as quadratic programming [41]. In the following section, we present how the proposed VLM-RRT approach generates the reference path $\mathcal{P}$ by integrating a Large Language Model with the Rapidly-exploring Random Tree algorithm. To simplify the analysis without loss of generality, we assume that the UAV operates at a fixed altitude, thereby restricting our formulation to a planar 2D setting. However, the proposed approach can be readily extended to three-dimensional navigation.

IV. Proposed Approach

Traditionally, we can obtain the reference path $\mathcal{P}$ using the RRT sampling-based path-planning algorithm, depicted in Alg. 1. As demonstrated, the algorithm receives as input the initial position of the agent, denoted as $y(0)$, at time $t = 0$, along with the goal region $\mathcal{G}_o \in \mathbb{R}^2$ (e.g., the centroid of the target region $\mathcal{G}$) and the set of fire fronts $\mathcal{O}$. The algorithm proceeds by incrementally constructing a tree V, originating from the agent's initial state and expanding toward the goal region $\mathcal{G}_o$. During each iteration, a point ν_{rand} is randomly sampled from the space, and the tree is extended from its closest existing vertex ν_{nearest} toward ν_{rand}, resulting in a new vertex ν_{new}, provided that the path does not intersect any fire fronts, ensuring navigational safety. The extension follows a predefined step size δ, systematically guiding the tree's exploration of the space. This iterative process continues until either the tree successfully reaches the goal region (i.e., $\|\nu_{\text{new}} - \mathcal{G}_o\|_2 \leq \epsilon$, where $\epsilon > 0$), in which case the path is retrieved via backtracking, or the predefined iteration limit N is reached, resulting in the algorithm failing to converge.

Algorithm 1 Traditional RRT Algorithm

Require: $y(0),\ \mathcal{G}_o,\ \mathcal{O},\ \delta$
$V \leftarrow \{y(0)\},\ E \leftarrow \emptyset,\ r \leftarrow \emptyset,\ i \leftarrow 0$
while $i < N$ **do**
  $\nu_{\text{rand}} \leftarrow \text{SampleState}()$
  $\nu_{\text{nearest}} \leftarrow \text{NearestNeighbor}(V, \nu_{\text{rand}})$
  $\nu_{\text{new}} \leftarrow \text{Steer}(\nu_{\text{nearest}}, \nu_{\text{rand}}, \delta)$
  $i \leftarrow i + 1$
  if $\text{PathFree}(\nu_{\text{new}}, \nu_{\text{nearest}}, \mathcal{O})$ **then**
    $V \leftarrow V \cup \{\nu_{\text{new}}\},\ E \leftarrow E \cup \{(\nu_{\text{nearest}}, \nu_{\text{new}})\}$
  end if
  if $||\nu_{\text{new}} - \mathcal{G}_o||_2 \leq \epsilon$ **then**
    $\mathcal{P} \leftarrow \text{RetrievePlan}(V, E, \nu_{\text{new}})$
    return $\mathcal{P}$
  end if
end while

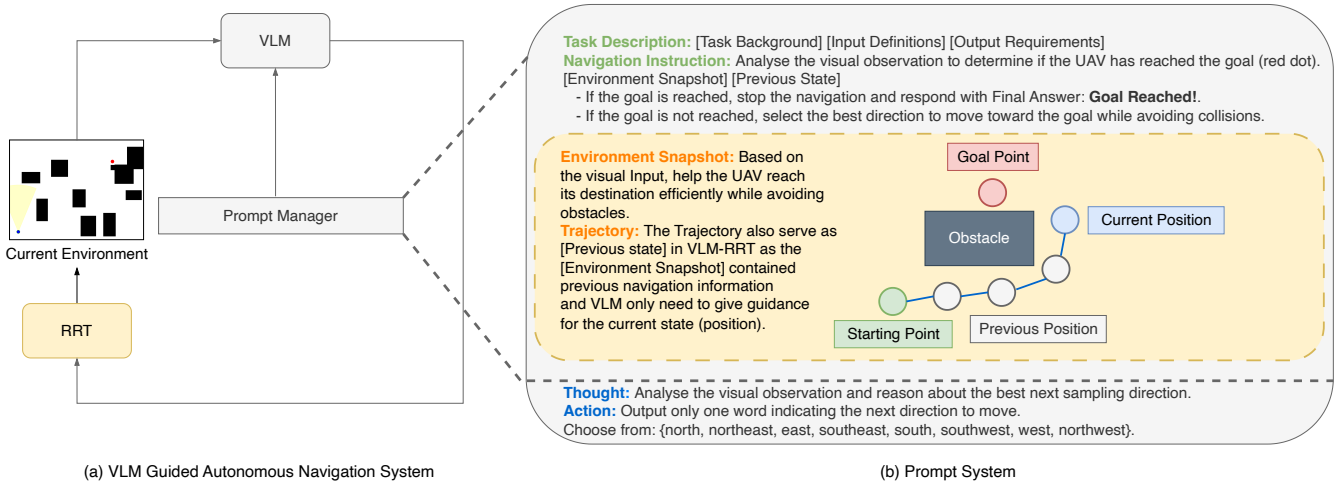
The RRT algorithm, while widely effective across various motion planning tasks, encounters several inherent challenges that can impact its performance. One significant limitation is low sampling efficiency, as traditional RRT often produces a high proportion of invalid or redundant samples. This inefficiency increases computational overhead and hampers the algorithm's ability to explore the search space effectively. Additionally, the resulting paths may include unnecessary detours or redundant nodes, which can lead to suboptimal path quality and increased traversal cost.

To address these limitations, VLM-RRT utilizes LLMs as general-purpose pattern-matching machines and integrates their reasoning capabilities into the RRT search to guide the sampling process toward regions most likely to contain the optimal solution.

A. VLM-RRT

The VLM-RRT algorithm, shown in Alg. 2, integrates VLMs into the sampling process of RRT. In this framework, VLMs serve as general-purpose pattern-matching and reasoning machines that extract contextual information from the current environment, as illustrated in Fig. 1. The VLM output aims to improve sampling efficiency and steer the tree exploration toward regions that are more likely to yield an optimal path. In our context, "optimal" refers to the collision-free route that minimizes the total travel distance from the starting position to the goal region, ensuring the most efficient navigation.

At the start of each planning iteration, the algorithm captures the current state of the environment as an image



(a) VLM Guided Autonomous Navigation System (b) Prompt System

Fig. 1. Our basic system consists of two types of prompts, task descriptions and basic inputs. We match a snapshot of the current environment with the task instructions, incorporating the current navigation state and history into the prompt to activate the agent's global dynamic exploration capability.

Algorithm 2 VLM-RRT Algorithm

Require: $y(0),\ \mathcal{G}o,\ \mathcal{O},\ \delta, \gamma$

1: $V \leftarrow y(0),\ E \leftarrow \emptyset,\ r \leftarrow \emptyset,\ i \leftarrow 0$
2: **while** $i < N$ **do**
3: $E_{\text{current}} \leftarrow \text{GetEnvironmentState}()$
4: $\alpha \leftarrow \mathcal{U}(0, 1)$
5: **if** $\alpha \leq \gamma$ **then**
6: $\hat{\nu} \leftarrow \text{PickLeafNode}(V, E_{\text{current}})$
7: $d \leftarrow \text{GetVLMdirection}(\hat{\nu}, E_{\text{current}})$
8: $\nu_{\text{rand}} \leftarrow \text{SampleStateVLM}(\hat{\nu}, d, r, \theta)$
9: **else**
10: $\nu_{\text{rand}} \leftarrow \text{SampleState}()$
11: **end if**
12: $\nu_{\text{nearest}} \leftarrow \text{NearestNeighbor}(V, \nu_{\text{rand}})$
13: $\nu_{\text{new}} \leftarrow \text{Steer}(\nu_{\text{nearest}}, \nu_{\text{rand}}, \delta)$
14: $i \leftarrow i + 1$
15: **if** $\text{PathFree}(\nu_{\text{new}}, \nu_{\text{nearest}}, \mathcal{O})$ **then**
16: $V \leftarrow V \cup \nu_{\text{new}},\ E \leftarrow E \cup (\nu_{\text{nearest}}, \nu_{\text{new}})$
17: **end if**
18: **if** $||\nu_{\text{new}} - \mathcal{G}_o||_2 \leq \epsilon$ **then**
19: $\mathcal{P} \leftarrow \text{RetrievePlan}(V, E, \nu_{\text{new}})$
20: **return** $\mathcal{P}$
21: **end if**
22: **end while**

E_{current} using the function `GetEnvironmentState`. This image encodes the locations of fire fronts $\mathcal{O}$, the goal region, and the state of exploration represented by the tree V. In this representation, the goal region and the leaf nodes in V are distinguished using different colors.

Subsequently, with probability γ, the algorithm decides whether to take a VLM-informed exploration step. In Line 4, the random variable α is drawn from the uniform distribution in the range $(0, 1)$. With probability γ, the proposed approach randomly selects a leaf node $\hat{\nu}$ from V, as shown in Line 6 of Alg. 2, and then employs the VLM to determine the direction d in which the agent should move to reach the goal region, given the selected node $\hat{\nu}$. This is achieved through the function `GetVLMdirection`, as shown in Line 7, which leverages the VLM via prompt engineering to detect the goal region and reason about the direction d the agent should take by analyzing the environment image E_{current}.

The algorithm then proceeds by randomly sampling a new point ν_{rand} from a sector $\mathcal{R}$, centered at $\hat{\nu}$, with direction d, radius r, and angle θ, as described in Line 8. Otherwise, with probability $1 - \gamma$, the algorithm follows the standard RRT sampling strategy, selecting ν_{rand} from anywhere in the environment using the function `SampleState`, as shown in Line 10. Subsequently, the algorithm operates similarly to the original RRT, where the tree is expanded from its nearest existing vertex, ν_{nearest}, in the direction of ν_{rand}, generating a new vertex ν_{new}. This expansion occurs only if the resulting path does not intersect any fire fronts, thereby ensuring safe navigation toward the goal region. This iterative process continues until either the tree successfully reaches the goal region or the algorithm reaches the predefined iteration limit.

B. Prompt Engineering

The prompt engineering methodology leverages the VLM's pattern-matching capabilities. As shown in Fig. 1, our system implements structured prompts that combine task descriptions with environmental snapshots, incorporating both current navigation states and historical data. The prompt structure defines specific input parameters, output constraints, and environmental context for navigation guidance.

As demonstrated in Fig. 2, we experimented with three prompting techniques. Zero-shot prompting enables direct decision-making using only task instructions and current state information. Few-shot prompting augments this by including predefined example scenarios, ranging from unobstructed paths to multi-obstacle configurations, which serve as reference cases for similar navigation contexts. We integrated Chain of Thought (CoT) prompting [42] to enhance

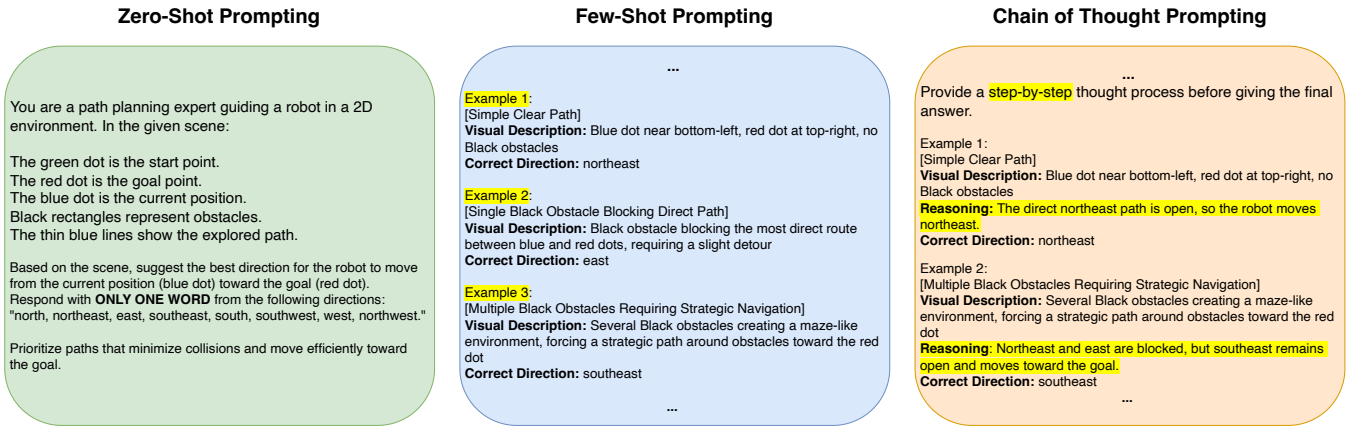


Fig. 2. Comparison of different prompt engineering techniques for navigation decision-making.

the system's reasoning capabilities. The CoT framework structures the decision-making process into explicit steps: obstacle identification, relative position analysis, and path feasibility evaluation. This structured approach enables the model to systematically process environmental constraints before determining movement directions.

V. Evaluation

A. Simulation Setup

To evaluate our approach, we assume that an autonomous UAV agent evolves (assuming a fixed altitude) inside a bounded environment $\mathcal{E} \subset \mathbb{R}^2$ of dimensions $500\,\text{m} \times 500\,\text{m}$. The agent's planar motion is captured by a 4-dimensional state vector $x(t) = [x_1, x_2, \dot{x}_1, \dot{x}_2]^\top \in \mathcal{X} \subset \mathbb{R}^4$, comprising its position $(x_1, x_2) \in \mathbb{R}^2$ and velocity $(\dot{x}_1, \dot{x}_2) \in \mathbb{R}^2$ components within the 2D Cartesian coordinate system. The agent is controllable and capable of following specific directional and speed commands via the control input $u(t) \in \mathcal{U} \subset \mathbb{R}^2$, which corresponds to the applied control force. The matrices $A \in \mathbb{R}^{4\times4}$ and $B \in \mathbb{R}^{4\times2}$, shown in Eq. (1), are given by:

$$A = \begin{bmatrix} I_{2\times2} & \Delta T \cdot I_{2\times2} \\ 0_{2\times2} & (1-\zeta) \cdot I_{2\times2} \end{bmatrix}, \quad B = \begin{bmatrix} 0_{2\times2} \\ \frac{\Delta T}{m} \cdot I_{2\times2} \end{bmatrix},$$

where ΔT signifies the sampling interval, ζ is the air resistance coefficient, and m represents the mass of the agent. Additionally, $I_{2\times2}$ and $0_{2\times2}$ are the 2-by-2 identity and zero matrices, respectively. The output vector $y(t) \in \mathcal{Y} \subset \mathbb{R}^2$ consists of the UAV's position at time step t; therefore, the matrices $C \in \mathbb{R}^{2\times4}$ and $D \in \mathbb{R}^{2\times2}$ are given by $\begin{bmatrix} I_{2\times2} & 0_{2\times2} \end{bmatrix}$ and $0_{2\times2}$, respectively. The parameters ΔT, ζ, and m are set to 1 s, 0.2, and 1.05 kg, respectively. The control input is bounded in each dimension within the range $[-10, 10]$ N, and the UAV's maximum velocity is capped at $v_{\max} = 15$ m/s. The starting and goal regions $\mathcal{S}$ and $\mathcal{G}$, as well as the fire fronts to be avoided $o \in \mathcal{O}$, are represented as rectangular regions with random dimensions, as shown in Fig. 1(a).

Unless otherwise indicated, the VLM-RRT step size δ is set to $\Delta T \cdot v_{\max}$ and $\epsilon = 1\,\text{m}$. The sampling sector $\mathcal{R}$ has radius $r = 30\,\text{m}$ and angle $\theta = 45°$, and the default value for the probability γ is set to 0.85. We should mention here that the output of the VLM-RRT algorithm is a path of length ℓ, i.e., a sequence of ℓ points, which is converted to the continuous reference path $\mathcal{P}$ to be tracked by fitting a spline curve [43]. The planning horizon is set to $T = 2.5\ell$, and subsequently, T evenly spaced points (i.e., with equal arc-length spacing) are sampled from $\mathcal{P}$ between the starting and goal points. The optimization in Eq. (2) is solved as a quadratic program (QP) with the Gurobi solver, with the matrices Q and R set to $0.9I_{2\times2}$ and $0.1I_{2\times2}$, respectively.

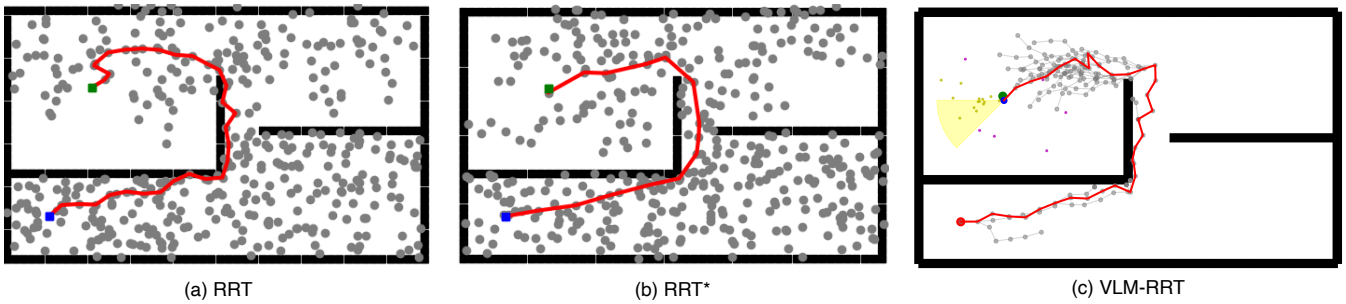
TABLE I
Performance Comparison with Competing Approaches.

Algorithm	LLM	Avg. Iterations (N)	Avg. Path Length (m)
A* [39]	-	514	52.73
LLM-A* [38]	GPT-4o	410	52.73
	Llama 3.2V	405	52.80
RRT	-	423	56.48
RRT* [28]	-	477	53.89
VLM-RRT	GPT-4o	172	54.56
	Llama 3.2V	176	55.87

We have evaluated the performance of our proposed approach by integrating two state-of-the-art VLMs to support autonomous navigation: OpenAI's GPT-4o [44] and Meta's Llama 3.2 90B Vision Instruct [45]. GPT-4o is a multimodal model with approximately 1.8 trillion parameters, accessed via OpenAI's API, whose advanced vision-language processing capabilities were leveraged for environmental interpretation and decision support within our UAV navigation framework. In contrast, Meta's Llama 3.2 90B Vision Instruct, with 90 billion parameters, facilitates multimodal reasoning to provide visual understanding and contextual analysis critical to the VLM-RRT path-planning algorithm.

B. Results

We begin the evaluation by comparing the proposed approach with the closely related work in [38], where the authors integrated LLMs with the A* path-finding algorithm [39]. Additionally, we compare it with the traditional RRT approach [23] and the RRT* algorithm [28]. Table I



(a) RRT (b) RRT* (c) VLM-RRT

Fig. 3. Illustrative example of the path-planning behavior obtained with: (a) RRT, (b) RRT* and (c) VLM-RRT.

TABLE II

PERFORMANCE OF VLM-RRT UNDER VARIOUS MODELS AND PROMPTING TECHNIQUES.

Algorithm	VLM	Prompt Technique	Success Rate	Avg. Iterations (N)	Avg. Path Length (m)
RRT	-	-	82% (41/50)	343	58
RRT*	-	-	88% (44/50)	302	45
VLM-RRT (Ours)	GPT-4o	Zero-shot	68% (34/50)	93	47
		Few-shot	94% (47/50)	89	46
		CoT	86% (43/50)	94	48
	Llama 3.2V	Zero-shot	56% (28/50)	102	48
		Few-shot	90% (45/50)	88	49
		CoT	78% (39/50)	95	47

presents the average number of iterations required for convergence and the resulting path length for each approach. These results were obtained by averaging 100 random scenarios (i.e., random environment configurations) in a Monte Carlo (MC) simulation. It is important to note that while the A* and RRT approaches are not directly comparable (in terms of the number of iterations), the results indicate a consensus on the performance improvement achieved when these planning algorithms are integrated with LLMs. In particular, the VLM-RRT approach significantly improves the convergence rate compared to the original RRT algorithm while also enhancing path quality (in terms of path length). Moreover, the VLM-RRT approach achieves path quality comparable to that of the more advanced RRT*, but with fewer iterations. An illustrative example is shown in Fig. 3. These results were obtained using CoT prompting.

In the next experiment, we conduct a more thorough analysis of the proposed VLM-RRT approach in terms of the following metrics:

1) **Success Rate:** Defined as the percentage of experiments in which a collision-free path from the start to the goal region is successfully found within the maximum number of iterations $N = 500$.
2) **Number of Iterations:** The total number of iterations required to obtain a feasible path.
3) **Path Length:** Assessed via the length of the final path from the start to the goal region.

The above metrics are computed on a per-experiment basis and then averaged across 250 MC runs, as shown in Table II. As shown in the results, RRT and RRT* achieve success rates of 82% and 88%, respectively. In contrast, VLM-RRT surpasses the performance of the traditional approaches, particularly under few-shot and CoT prompting, while requiring significantly fewer iterations. Additionally, VLM-RRT achieves higher path quality in terms of path length compared to the traditional RRT, as shown in Table II. In terms of path length, VLM-RRT achieves performance comparable to the more advanced RRT*, though this is accomplished with fewer iterations. The results also highlight the differences in performance obtained using different prompting techniques, as well as the effectiveness of the two LLMs, indicating an advantage for GPT-4o.

TABLE III

VLM-RRT ROBUSTNESS ANALYSIS

γ	Success Rate (%)	Avg. Number of Iterations (N)
1.0	79	86
0.9	88	92
0.8	92	105
0.7	95	102
0.6	93	128
0.5	89	142

The next experiment investigates how the parameter γ (i.e., the probability of taking a VLM-informed decision) affects the sensitivity of the algorithm and how this can be fine-tuned to increase the robustness of VLM-RRT. Table III shows the algorithm's performance (in terms of success rate and number of iterations) for different values of γ ranging from 1 to 0.5, obtained over 100 MC trials. When $\gamma = 1$, VLM-RRT always takes a VLM-informed decision at each time step and samples a new point from within the area suggested by the VLM. Although this can lead to faster convergence in many situations, the results show that this strategy decreases robustness (i.e., increases the likelihood

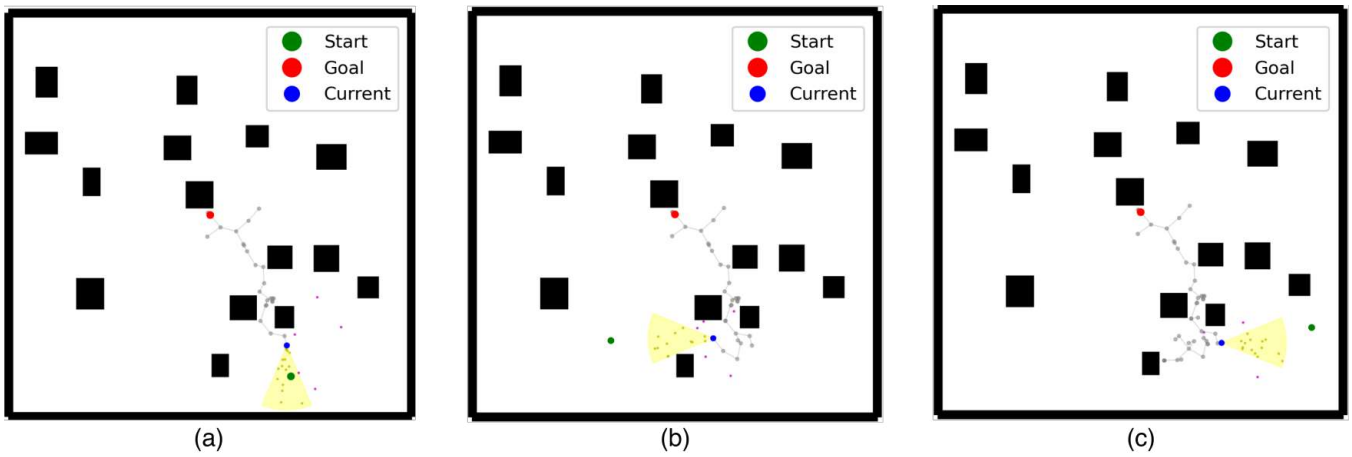


Fig. 4. Illustration of the VLM-RRT algorithm navigating toward dynamic goals in a 2D environment. The scenario involves three goal relocations. Red point is the starting position, green point is the goal position which changes over time, and the blue point is a leaf node. The yellow sector is the VLM-informed sampling region.

of failure to converge). From our experiments, we have observed that this is due to two main reasons: (a) the VLM can make mistakes, which consequently lead the algorithm to make incorrect decisions, and (b) $\gamma = 1$ leads to greedy behavior, which in turn causes the algorithm to become stuck in infeasible regions. On the other hand, a lower value of γ causes the algorithm to behave more similarly to the original RRT, resulting in a drop in success rate due to reaching the maximum number of iterations. Therefore, a fine-tuned value of γ optimally balances exploration and exploitation, leading to enhanced robustness and performance, as shown.

In emergency response scenarios, mission parameters often change dynamically as new information becomes available. For instance, the location of survivors may be updated based on new sensor data or witness reports, requiring rapid replanning of UAV trajectories. To evaluate our system's performance in such dynamic scenarios, we conducted a series of experiments with dynamic goal locations. Figure 4 shows one such scenario where the goal region changes location dynamically during the mission. The VLM demonstrated a consistent ability to identify changes in the location of the goal region during the mission, achieving a detection rate of 92% across 50 random scenarios in which the goal region's location varied over time. In cases where the VLM failed to immediately recognize the new goal location, it typically required one additional sampling iteration to correct its direction. We observed that the VLM's performance in recognizing and adapting to new goal locations remained robust even in cluttered environments, though the convergence time increased when obstacles were present between the UAV's position and the new goal location. This increase in convergence time was primarily due to the necessary local path adjustments around obstacles rather than any delay in goal recognition or sampling direction updates.

VI. Conclusion and Future Work

In this work, we present Visual-Language Model RRT (VLM-RRT), a hybrid path-planning framework that combines the pattern recognition capabilities of Vision-Language Models (VLMs) with the efficiency of Rapidly-exploring Random Trees (RRT). By utilizing VLMs to extract high-level semantic information from environmental snapshots, our approach directs sampling toward regions with a higher likelihood of containing feasible paths. This targeted bias significantly enhances both sampling efficiency and path quality. Our experimental evaluation demonstrates notable improvements in navigation performance compared to conventional sampling-based methods, highlighting the advantage of integrating VLM-based perception-driven guidance into motion planning. Future work will explore tighter integration between large language models (LLMs) and path-planning algorithms, focusing on how recent advances in reasoning models can further enhance autonomous decision-making and planning capabilities.

Acknowledgments

This work is implemented under the Border Management and Visa Policy Instrument (BMVI) and is co-financed by the European Union and the Republic of Cyprus (GA:BMVI/2021-2022/SA/1.2.1/015), and supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 739551 (KIOS CoE), and through the Cyprus Deputy Ministry of Research, Innovation and Digital Policy of the Republic of Cyprus.

References

[1] C. Vitale, S. Papaioannou, P. Kolios, and G. Ellinas, "Autonomous 4D trajectory planning for dynamic and flexible air traffic management," *Journal of Intelligent & Robotic Systems*, vol. 106, no. 1, p. 11, 2022.

[2] ——, "Probabilistically robust trajectory planning of multiple aerial agents," in *2024 18th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE, 2024, pp. 852–859.

[3] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[4] S. Papaioannou, P. Kolios, T. Theocharides, C. G. Panayiotou, and M. M. Polycarpou, "Distributed search planning in 3-d environments with a dynamically varying number of agents," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 7, pp. 4117–4130, 2023.

[5] ——, "Jointly-optimized Trajectory Generation and Camera Control for 3D Coverage Planning," *IEEE Transactions on Mobile Computing*, 2025, doi:10.1109/TMC.2025.3551362.

[6] ——, "Rolling horizon coverage control with collaborative autonomous agents," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 383, no. 2289, 2025.

[7] Y. Wan, Y. Zhong, A. Ma, and L. Zhang, "An accurate uav 3-d path planning method for disaster emergency response based on an improved multiobjective swarm intelligence algorithm," *IEEE Transactions on Cybernetics*, vol. 53, no. 4, pp. 2658–2671, 2022.

[8] S. Papaioannou, P. Kolios, T. Theocharides, C. G. Panayiotou, and M. M. Polycarpou, "3D Trajectory Planning for UAV-based Search Missions: An Integrated Assessment and Search Planning Approach," in *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2021, pp. 517–526.

[9] ——, "Towards Automated 3D Search Planning for Emergency Response Missions," *Journal of Intelligent & Robotic Systems*, vol. 103, no. 1, p. 2, 2021.

[10] S. Papaioannou, S. Kim, C. Laoudias, P. Kolios, S. Kim, T. Theocharides, C. Panayiotou, and M. Polycarpou, "Coordinated CRLB-based control for tracking multiple first responders in 3D environments," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2020, pp. 1475–1484.

[11] S. Papaioannou, P. Kolios, C. G. Panayiotou, and M. M. Polycarpou, "Synergising human-like responses and machine intelligence for planning in disaster response," in *2024 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2024, pp. 1–8.

[12] S. Papaioannou, P. Kolios, and G. Ellinas, "Distributed estimation and control for jamming an aerial target with multiple agents," *IEEE Transactions on Mobile Computing*, vol. 22, no. 12, pp. 7203–7217, 2022.

[13] Y. Wu, S. Wu, and X. Hu, "Cooperative path planning of uavs & ugvs for a persistent surveillance task in urban environments," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4906–4919, 2020.

[14] S. Papaioannou, P. Kolios, T. Theocharides, C. G. Panayiotou, and M. M. Polycarpou, "Probabilistic search and track with multiple mobile agents," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 253–262.

[15] ——, "A cooperative multiagent probabilistic framework for search and track missions," *IEEE Transactions on Control of Network Systems*, vol. 8, no. 2, pp. 847–858, 2020.

[16] S. Papaioannou, C. Laoudias, P. Kolios, T. Theocharides, and C. G. Panayiotou, "Joint estimation and control for multi-target passive monitoring with an autonomous UAV agent," in *2023 31st Mediterranean Conference on Control and Automation (MED)*. IEEE, 2023, pp. 176–181.

[17] W. Jing, D. Deng, Y. Wu, and K. Shimada, "Multi-uav coverage path planning for the inspection of large and complex structures," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1480–1486.

[18] Y. Sun and O. Ma, "Automating aircraft scanning for inspection or 3d model creation with a uav and optimal path planning," *Drones*, vol. 6, no. 4, p. 87, 2022.

[19] S. Papaioannou, P. Kolios, T. Theocharides, C. G. Panayiotou, and M. M. Polycarpou, "UAV-based receding horizon control for 3D inspection planning," in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2022, pp. 1121–1130.

[20] S. Papaioannou, C. Vitale, P. Kolios, C. G. Panayiotou, and M. M. Polycarpou, "Hierarchical Fault-Tolerant Coverage Control for an Autonomous Aerial Agent," *IFAC-PapersOnLine*, vol. 58, no. 4, pp. 532–537, 2024, 12th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS 2024.

[21] S. Papaioannou, P. Kolios, T. Theocharides, C. G. Panayiotou, and M. M. Polycarpou, "Cooperative receding horizon 3D coverage control with a team of networked aerial agents," in *2023 62nd IEEE Conference on Decision and Control (CDC)*. IEEE, 2023, pp. 4399–4404.

[22] ——, "Unscented optimal control for 3D coverage planning with an autonomous UAV agent," in *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2023, pp. 703–712.

[23] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Computer Science Department, Iowa State University*, 1998.

[24] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.

[25] O. Arslan and P. Tsiotras, "Machine learning guided exploration for sampling-based motion planning algorithms," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 2646–2652.

[26] J. Wang, X. Jia, T. Zhang, N. Ma, and M. Q.-H. Meng, "Deep neural network enhanced sampling-based path planning in 3d space," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 3434–3443, 2021.

[27] Y.-L. Kuo, A. Barbu, and B. Katz, "Deep sequential models for sampling-based planning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 6490–6497.

[28] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[29] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *2014 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2014, pp. 2997–3004.

[30] S. Huang, "Path planning based on mixed algorithm of rrt and artificial potential field method," in *2021 4th International Conference on Intelligent Robotics and Control Engineering (IRCE)*. IEEE, 2021, pp. 149–155.

[31] X. Li and Y. Tong, "Path planning of a mobile robot based on the improved rrt algorithm," *Applied Sciences*, vol. 14, no. 1, 2024.

[32] L. Liu, Y. Zhang, L. Zhang, and M. Kermanshabi, "Rrt-cbf based motion planning," *arXiv preprint arXiv:2410.00343*, 2024.

[33] Z. Huang, H. Chen, J. Pohovey, and K. Driggs-Campbell, "Neural informed rrt*: Learning-based path planning with point cloud state representations under admissible ellipsoidal constraints," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 8742–8748.

[34] J. Wang, W. Chi, C. Li, C. Wang, and M. Q.-H. Meng, "Neural rrt*: Learning-based optimal path planning," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 1748–1758, 2020.

[35] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[36] T. B. Brown, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[37] G. Zhou, Y. Hong, and Q. Wu, "Navgpt: Explicit reasoning in vision-and-language navigation with large language models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 7, 2024, pp. 7641–7649.

[38] S. Meng, Y. Wang, C.-F. Yang, N. Peng, and K.-W. Chang, "Llm-a*: Large language model enhanced incremental heuristic search on path planning," *arXiv preprint arXiv:2407.02511*, 2024.

[39] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968. [Online]. Available: https://doi.org/10.1109/tssc.1968.300136

[40] J. P. How, E. Frazzoli, and G. V. Chowdhary, "Linear flight control techniques for unmanned aerial vehicles," in *Handbook of unmanned aerial vehicles*. Springer, 2015, pp. 529–576.

[41] J. Nocedal and S. J. Wright, "Quadratic programming," *Numerical optimization*, pp. 448–492, 2006.

[42] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[43] H. Wang, J. Kearney, and K. Atkinson, "Arc-length parameterized spline curves for real-time simulation," in *Proc. 5th International Conference on Curves and Surfaces*, vol. 387396, 2002.

[44] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[45] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.