Look Further Ahead: Testing the Limits of GPT-4 in Path Planning

Mohamed Aghzal[1] **Erion Plaku** [2*] **Ziyu Yao** [1]

[1]George Mason University [2]National Science Foundation

Abstract

Large Language Models (LLMs) have shown impressive capabilities across a wide variety of tasks. However, they still face challenges with long-horizon planning. To study this, we propose *path planning* tasks as a platform to evaluate LLMs' ability to navigate long trajectories under geometric constraints. Our proposed benchmark systematically tests path-planning skills in complex settings. Using this, we examined GPT-4's planning abilities using various task representations and prompting approaches. We found that framing prompts as Python code and decomposing long trajectory tasks improve GPT-4's path planning effectiveness. However, while these approaches show some promise toward improving the planning ability of the model, they do not obtain optimal paths and fail at generalizing over extended horizons.

INTRODUCTION

Trained on vast amounts of data, Large language models (LLMs) have demonstrated outstanding performance across a wide spectrum of tasks (OpenAI et al. 2023; Touvron et al. 2023; Wei et al. 2022; Chen et al. 2023b). However, these models still struggle on tasks requiring end-to-end planning and long-horizon reasoning (Valmeekam et al. 2023a,b; Yang and Tomar 2023), which are fundamental for their applications to robotics.

To facilitate the assessment of LLMs' planning capabilities, *path planning* has emerged as a promising venue in recent years. It involves determining a viable route for an agent to move from a starting point to a goal location while avoiding obstacles. Hence, it offers a straightforward yet challenging environment for testing grounding and long-horizon planning problems, making it highly relevant to various robotics applications. Prior benchmarks for path planning include BabyAI (Chevalier-Boisvert et al. 2019) and gSCAN (Ruis et al. 2020; Qiu et al. 2021); however, these datasets were proposed mainly for studying linguistic understanding in grounded environments and the planning settings are relatively simple. For example, since their settings are on relatively small grids (e.g., 6 by 6), the tasks can typically be solved within a small number of steps. Moreover, as they consist of only randomly scattered obstacles, their environments are not representative of a real-world navigation problem. In such settings, the models can often serendipitously find a path from the expansively unblocked space that evades obstacles, rather than developing a reliable strategy for obstacle avoidance.

To address these limitations, we propose a new benchmark aiming to more reliably assess the path-planning ability of LLMs. In particular, we target environments with larger grid sizes (i.e., 25 by 25) and with more geometric constraints, such as those shown in Fig. 2. The synthetic nature of our benchmark and its flexible experimental setup allows for the easy generation of novel settings. It can, thus, serve as a valuable resource for future research on the path-planning capabilities of LLMs.

Our benchmark and experiments provide insights into the following research questions (RQs):

- RQ1: Can LLMs be used to effectively plan paths in complex geometric environments?
- RQ2: How should the environments be represented?
- RQ3: How should the LLMs be prompted?

Answering RQ1 requires addressing the more fundamental challenges of RQ2 and RQ3. Specifically, RQ2 targets the foremost challenge to leverage LLMs for path planning, i.e., how to describe the task environment to the models (called "task representation").

The most natural way could be to employ large multimodal models (LMMs), such as GPT-4V (OpenAI 2023), and feed a snapshot of the environment as the task representation. However, state-of-the-art LMMs have been found to have extensive perceptual errors (Mitchell, Palmarini, and Moskvichev 2023; Tong et al. 2024; Yue et al. 2023). In our preliminary experiments, GPT-4V was unable to understand the original task environment. This weakness in perception, thus, introduces confounding variables that hinder our analysis of LMMs' path-planning capability. Conversely, directly verbalizing all of the obstacles in a complex environment (called "naive enumeration") is non-optimal, as it easily leads to overly long prompts, which may not be easy for an LLM to digest. Observing this challenge, our work first explores two novel representations, i.e., "code rep-

*The work by E. Plaku is supported by (while serving at) the National Science Foundation. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

resentation", which uses a Python code snippet to describe the process of locating the obstacles on the grid, and "grid representation", which is a 2-dimensional string representation of the full environment (Fig. 1). Intuitively, the code representation allows for a more compact yet unambiguous way to describe the environment, while the grid representation aligns more with human intuition and may thus help the LLM planning.

LLMs have shown varying levels of performance depending on how they are prompted (Wei et al. 2022; Chen et al. 2023b; Mueller et al. 2023). Therefore, our RQ3 looks into the more effective ways to prompt an LLM for path planning. Specifically, we consider the naive few-shot prompting (Brown et al. 2020) as a baseline. Prior work showed that for an LLM to fully conceptualize a complex environment, it is crucial to let it directly interact with the environment and build a mental image of the space based on the environment feedback (Yao et al. 2023b). We generalize the idea to the novel setting of path planning and propose "Planning with Feedback", a prompting approach that allows an LLM to execute its partial action sequence, observe the outcome, and adjust its plan dynamically. Finally, considering the challenge of planning for a long trajectory, we also propose "Task Decomposition", a prompting approach that decomposes the long-range problem into smaller shorter segments and then prompts the LLM to complete each of them one by one. This approach was found helpful in prior work (Zhou et al. 2023; Prasad et al. 2023; Khot et al. 2023) but has not been tested in path planning.

RELATED WORK

LLMs as Autonomous Planning Agents

Using LLMs to perform planning has emerged as a prominent theme across several recent studies. For instance, the work in (Huang et al. 2022a) highlighted the potential of LLMs to serve as planning agents, and SayCan (Ahn et al. 2022) used LLMs to transform natural language instructions into actionable plans for robotic applications. However, several studies argued that LLMs are not well suited for end-to-end planning tasks (Valmeekam et al. 2023b,a; Pallagani et al. 2023; Chen et al. 2023a), despite others demonstrating that they could be enhanced when augmented with tree search (Hao et al. 2023; Zhao, Lee, and Hsu 2023; Yao et al. 2023a) or used with a classic planner (Chen et al. 2023c; Liu et al. 2023; Silver et al. 2023; Kambhampati et al. 2024; Xie et al. 2023).

Another promising idea to enhance LLM planning is to leverage environmental feedback (Yao et al. 2023b; Song et al. 2023; Wang et al. 2023; Sun et al. 2023; Raman et al. 2022; Huang et al. 2022b). We generalize this idea in the context of path planning and look at how it scales up to plans that require a longer number of actions.

Furthermore, while natural language may be the most intuitive method for prompting LLMs, it may not always be an optimal representation for unleashing their full capability. For instance, in (Lin et al. 2023), the authors demonstrated that using a table representation yields superior performance on embodied planning tasks, and in (Madaan et al. 2022; Mueller et al. 2023; Chen et al. 2023b; Puerto et al. 2024), authors found that code representations could better elicit the reasoning capability in LLMs. In this same spirit, we experiment with a novel Python code representation for path planning, which offers a compact and unambiguous way to describe the environments as well as the tasks that ought to be solved. Our observation is consistent with recent work, which showed the advantage of code representations.

To the best of our knowledge, this work presents the first exploration of code representation for path planning.

Benchmarks for LLM Path Planning

The potential of LLMs in navigation tasks has been a topic of interest in recent years. Several embodied datasets (Anderson et al. 2018; Gordon et al. 2018; Shridhar et al. 2020) have been proposed in the past. However, these datasets introduced additional confounding variables (i.e. vision component), which may affect the LLM performance. Text-only embodied navigation benchmarks (Shridhar et al. 2021; Côté et al. 2018) have also been introduced; nevertheless, the planning required to solve the tasks involves merely planning over short horizons.

Conducting path and motion planning with LLMs has gained traction recently (Xiao and Wang 2023; Chen et al. 2023c; Ding et al. 2023; Chen, Koenig, and Dilkina 2024). To this end, several benchmarks have been proposed. For instance, datasets such as BabyAI (Chevalier-Boisvert et al. 2019) and gSCAN (Ruis et al. 2020; Qiu et al. 2021) were proposed to study grounded language learning through 2D navigation tasks, however, the focus on these tasks was on linguistic generalization and task understanding, whereas the planning problems considered are simple and may not be reflective of the limits to which LLMs can be pushed in terms of path planning. In a recent technical report (Aghzal, Plaku, and Yao 2023), we proposed a benchmark specifically designed for path planning. However, the task environments we considered there were simplistic, consisting of only random obstacle placements in a small grid size, which is not reflective of real-world applications. Our work in this paper fills the gap by proposing a new benchmark dataset with more complex geometric shapes and larger grid sizes. Under such more realistic task environments, we systematically explored different task representations and prompting approaches for utilizing LLMs for path planning. As such, we expect our benchmark and experimental results to inspire future research on this topic.

MATERIALS AND METHODS

Benchmark Data Synthesis

Geometric Environments For our main experiments, we use $N \times N$ grid environments, where $N = 25$. As shown in Fig. 2, we create three types of environments: (a) square mazes, where the agent must navigate through squares with one opening each and find the correct entrances; (b) rectangular blocks, where the agent faces diverse and irregular obstacles that block its path; and (c) zig-zag mazes, where the agent has to locate the opening on each horizontal wall and

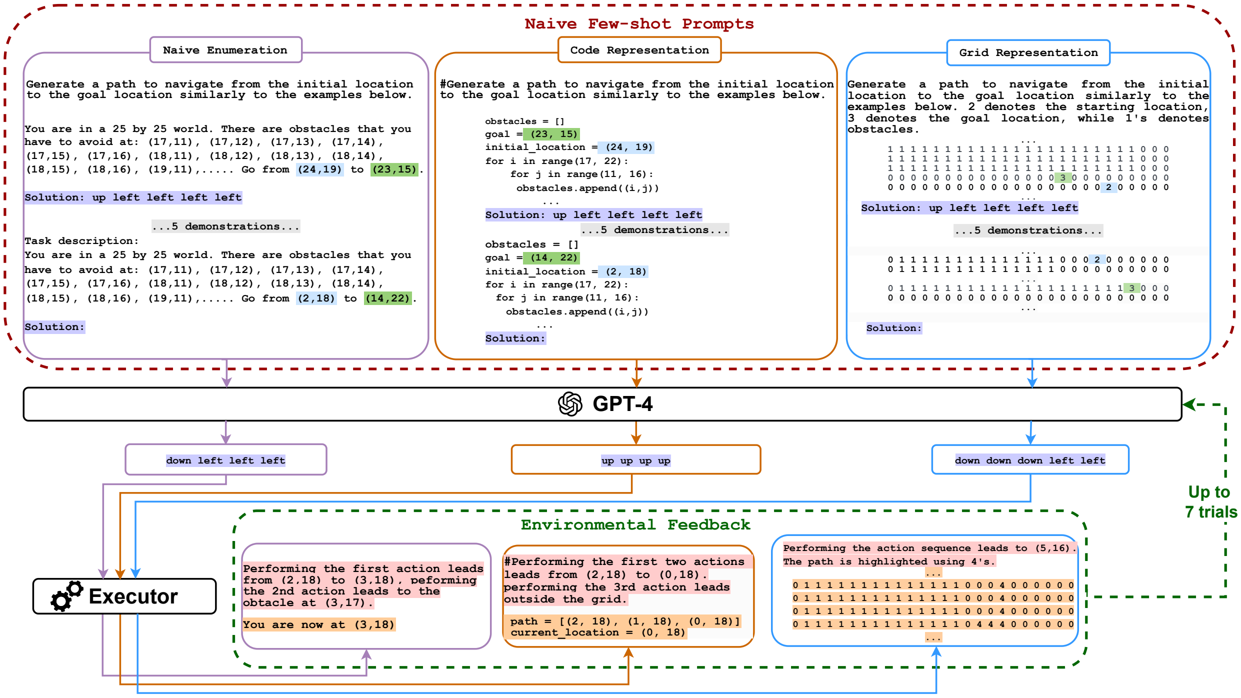
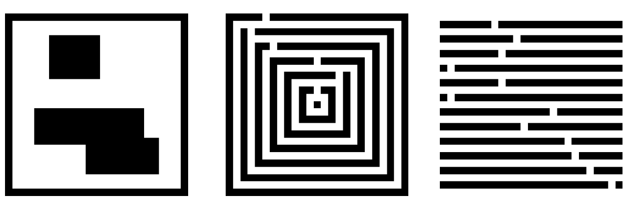


Figure 1: Overview of our *planning with feedback* prompting method using the different representations. The example shown is of a rectangular blocks setting. Solutions are highlighted in purple, initial locations are highlighted in blue, while goal locations are highlighted in green. Environmental feedback consists of a warning message (highlighted in pink) explaining the cause behind failure, and the current status of the agent after performing the actions (highlighted in orange).



(a) (b) (c)

Figure 2: Summary of the different environment types used in the experiments. The black regions represent the obstacles (walls), while the white space represents free cells. The figure shows one instance from each of the three types: ***(a) rectangular blocks***, where certain regions are completely blocked, ***(b) square mazes***, alternating squares with a single opening on each square, and ***(c) zig-zag mazes***, consisting of horizontal obstacles on alternating rows except one opening.

make frequent turns to reach the goal. For each environment type, we randomly sample a set of *30* different instances.

These environments provide challenging planning tasks for LLMs as they have to navigate in narrow passages, avoid large obstacles, make frequent turns, and take many steps to reach the goal. This allows us to assess LLMs' ability for long-range planning in complex, obstacle-rich, environments.

Ground-Truth Plan Generation We also aim to assess an LLM's ability for length generalization in the context of path planning (i.e., the ability of LLMs to succeed on paths requiring longer sequences than the demonstrations shown to them). Accordingly, we generate our planning scenarios of varying lengths. We adjust these values based on the specific geometries and the maximum path lengths they allow. We sample each of the path lengths in Table 1 once from each environment. We generate the paths for our ground-truth solutions using the A^* algorithm (Hart, Nilsson, and Raphael 1968). We designate instances of each path length as *In-distribution (IID)* , i.e., instances of shorter path lengths similar to the demonstrations shown to the LLM, or *Out-of-distribution (OOD)*, i.e., tasks involving longer-range planning compared to the demonstrations observed by the LLM.

Task Representations

Representing complex task environments as prompts for LLMs is challenging. Prior research overlooked this when focusing on small planning tasks. To understand the impact of task representation, we analyze three representations (Fig. 1):

- ***Naive Enumeration:*** This is the naive baseline which simply lists all of the obstacles on the grid. As a result, it often leads to very long prompts, making it difficult for an LLM to understand the task.

Table 1: Data Overview: We randomly chose one start and goal pair for every path length across all environments. In total, we sampled **150 IID** and **150 OOD** instances per geometric setting.

Geometry	**# Env.**	**Path Length Values**	
		IID	**OOD**
Rect. blocks	30	2, 5, 10, 15, 20	25, 30, 35, 40, 45
Square Mazes	30	5, 10, 15, 20, 25	30, 40, 50, 60, 75
Zig Zag	30	2, 5, 10, 15, 20	30, 50, 60, 75, 100

- ***Code Representation:*** LLMs have shown promise in a variety of tasks when prompted using code chen2023program, mueller2023incontext, madaan2022language, puerto2024code. Hence, we assess LLMs ability to conduct path planning when the task specification is provided using a description of the setting in Python code. To this end, we define variables specifying the start and goal locations as well as the logic to place the obstacles on the grid to form the geometric shape portrayed in the environment. Intuitively, code can offer a compact yet unambiguous way to define the task setting.
- ***Grid Representation:*** Humans find grid tasks easier with visual representation. Inspired by this, we evaluated an LLM's planning using grids where 1's denote obstacles, 2 indicates the start, and 3 marks the goal.

Prompting Methodologies

LLMs have shown great ability in learning from few-shot demonstrations, giving rise to a novel paradigm known as in-context learning (Brown et al. 2020). However, prior work also found that LLMs can be sensitive to the specific way how these few-shot demonstrations are designed (Wei et al. 2022; Chen et al. 2023b; Mueller et al. 2023). In experiments, we compare a total of three prompt designs to understand the potential of LLMs being prompted for path planning.

- ***Naive Few-Shot***: We explored the naive few-shot prompting approach from (Brown et al. 2020), where an LLM is prompted with a few examples of tasks and their correct action sequences. The model was given five demonstrations from the same environment as the test instance, using IID-sampled values.
- ***Planning with Feedback***: Environmental feedback has been shown to enhance the planning capabilities of LLMs (Yao et al. 2023b; Sun et al. 2023; Song et al. 2023). We generalize this idea to path planning by initially prompting the LLM to generate a plan. Subsequently, when a failure is about to occur, we supply feedback at the failure point, encouraging the model to continue its planning from that juncture. The "feedback" considered in our experiment is a natural language sentence indicating how an LLM's next action will lead to an obstacle (Fig. 1), which simulates how a physical robot's local sensor could emit a warning message when the robot is detected to be close to an obstacle. We allow up to 7 trials as a trade-off between thorough exploration of potential solutions and preventing infinite loops and/or high inference costs.
- ***Task Decomposition***: Recent work (Valmeekam et al. 2023b; Aghzal, Plaku, and Yao 2023; Valmeekam et al. 2023a; Yang and Tomar 2023) has shown the shortcomings of LLMs in long-horizon planning. On the other hand, several papers have shown that LLMs' success on complex tasks can be improved by decomposing them into smaller, simpler sub-tasks (Khot et al. 2023; Prasad et al. 2023). As such, we assess GPT-4's strength in navigation over short horizons by evaluating how it performs if we decompose a long-range planning problem into multiple simpler problems. Accordingly, we reduce planning problems into sub-tasks consisting of 5 or fewer steps. This is achieved by decomposing the ground-truth solution into sub-steps, providing the LLM with pairs of initial and goal locations of each sub-problem, and assessing whether it can solve all of the sub-problems.

Finally, we note that the popular approach of Chain-of-Thought (CoT) (Wei et al. 2022), though effective in short-horizon planning tasks, is not practical in our context. As the tasks in our setting require reasoning over long trajectories, this step-by-step reasoning becomes both costly and inaccurate.

Model and Implementation

We experiment with GPT-4 using a variety of prompting techniques and representations. We access the "gpt-4-turbo" version of the model through the OpenAI API.[1] We set the temperature to 0 to encourage the results to be reproducible. In addition, we limit the generation output to 200 tokens for all experiments. We provide our code and prompt examples on GitHub to enable experiment replication.[2] Our benchmark to designed to be extensible, accommodating new geometric settings, for researchers wishing to further explore the topic.

Evaluation

We evaluate the performance of an LLM in path planning using the following metrics: ***(1) Success Rate (%)***, which measures the proportion of paths that successfully navigate from the starting point to the designated goal. We note that for this case, if the goal is reached before executing the full path, then it is marked as a success; ***(2) Optimal Rate (%)***, representing the proportion of paths that are of the same length as the ground truths calculated using A^* (Sec.); ***(3) Exact***

[1]https://openai.com/blog/openai-api

[2]Our code, data and prompt examples can be found on the following link

Match Accuracy (%), the proportion of paths that *precisely match* the ground-truth plan calculated in advance. Note that for the two maze environments, exact match accuracy always equals the optimal rate. However, for a rectangular block environment, there could exist multiple optimal paths, hence its Exact Match Accuracy is a more strict metric than its Optimal Rate.

RESULTS AND ANALYSIS

In Fig. 3, we present the results when different prompt methodologies are combined with various task representations when prompting an LLM for path planning.

Planning with Different Task Representations

Describing tasks using code is promising: GPT-4 generally performs better when prompted with the code representation. This is consistent with previous work, which suggests that LLMs can conduct better reasoning when prompted using code (Mueller et al. 2023; Chen et al. 2023b). The compactness of the code representation can also be used to explain this improvement. As shown in Table 2, code can provide the task specification using significantly fewer input tokens, when compared to naive enumeration. Nevertheless, this method has a drawback: the requirement for manually designing a template to describe tasks according to the depicted geometry.

Naive enumeration falls short: Having to list all of the obstacles can lead to long prompts, which, in turn, results in a longer context window for the model, which hinders LLM performance. Performance in square maze environments is the lowest with naive enumeration, due to the higher number of obstacles, as Table 2 indicates.

LLMs fail to conceptualize 2-dimensional grids: Performance with grid representation was notably poor, with GPT-4 often generating random sequences that failed to direct the LLM agent correctly. This is contrary to human intuitions as we often prefer developing an overview image of the environment before making a plan. This issue may stem from LLMs' sequential input processing, making the two-dimensional task specification ill-suited for LLMs.

Planning in Different Geometries

Planning is easier in rectangular block environments: Fig. 3 showcases superior performance in terms of success rate on rectangular block environments across all representations. This type of environment was easier to navigate for the agent. This is because the environments under this design are typically less complex, and multiple paths can be taken to reach the goal. The two maze environments were harder for the agent to navigate, across all representations. This highlights that environmental complexity plays an important role in the LLMs capability to plan.

Long-horizon planning is more difficult in complex environments: The lower performance in the two maze environments offers insights into what decides the difficulty of a "planning" task. For instance, navigating 10 steps horizontally is not necessarily a more difficult task than a scenario involving moving 5 steps to reach a goal two levels down *(e.g. left down right right down)*. This further highlights the need for evaluating LLMs planning in cases that pose a challenge not solely from a temporal planning axis, but also under different geometric settings. We notice that GPT-4's performance drops more rapidly in zig-zag environments. This can be explained by the nature of navigation in this environment, which typically requires making more frequent turns to go from the initial to the goal locations. Task decomposition often fails on the sub-tasks for making such turns. This highlights GPT-4's shortcomings in dealing with complex geometries, even in short-sighted scenarios.

Length Generalization with Different Prompt Methods

GPT-4 struggles to strategize over long-horizon paths: In Fig. 3, we can observe a drop in performance as we increase the path lengths. This highlights GPT-4's inability to plan over longer trajectories. Reducing the long planning problem into smaller sub-segments helps improve generalization in rectangular block environments because problem decomposition in this case leads to simpler geometries. As the rectangular blocks form random regions across the grid, oftentimes, the optimal ground-truth paths are across regions consisting of mostly free space. Exposing points from such a plan prompts the model to solve sub-tasks involving fewer obstacles. Decomposition based on length does not achieve this in the two maze environments; as the obstacles under these settings are evenly distributed across the grid.

GPT-4 shows promise as a short-sighted planning agent: Task decomposition showcases enhanced performance compared to the other methods on long trajectory scenarios. This showcases LLMs' ability to solve short-sighted planning tasks in our environments. This highlights the potential for incorporating GPT-4 in frameworks that require the LLM to conduct localized decision-making.

Feedback is useful, particularly in rectangular blocks: Allowing GPT-4 to interact with the environment and observe the effect of its actions shows promise, particularly in rectangular block environments. This showcases that GPT-4 can guide the agent in the correct "general" direction and can recover by providing a new plan in case it encounters an illegal action. Nevertheless, this technique still fails on OOD path lengths. The success in shorter tasks is a result of implicitly solving multiple smaller subproblems. Longer-horizon tasks would require breaking the problem down into more than seven subtasks (i.e. more than 7 interactions with the environments). As such, increasing this value may offer improvements, but this can incur high inference costs.

Optimal Planning

GPT-4 is unable to find the optimal strategy: As can be seen from the *optimal rate* metric, the LLM struggles to find the optimal path in almost all instances. Upon examining the model's outputs, we notice that it opts for unnecessarily long trajectories, even in cases where the goal is within close range. Curiously, a common trend across the paths adopted by the model in successful cases tends to resemble a back-tracking approach where the agent tasks several steps in a

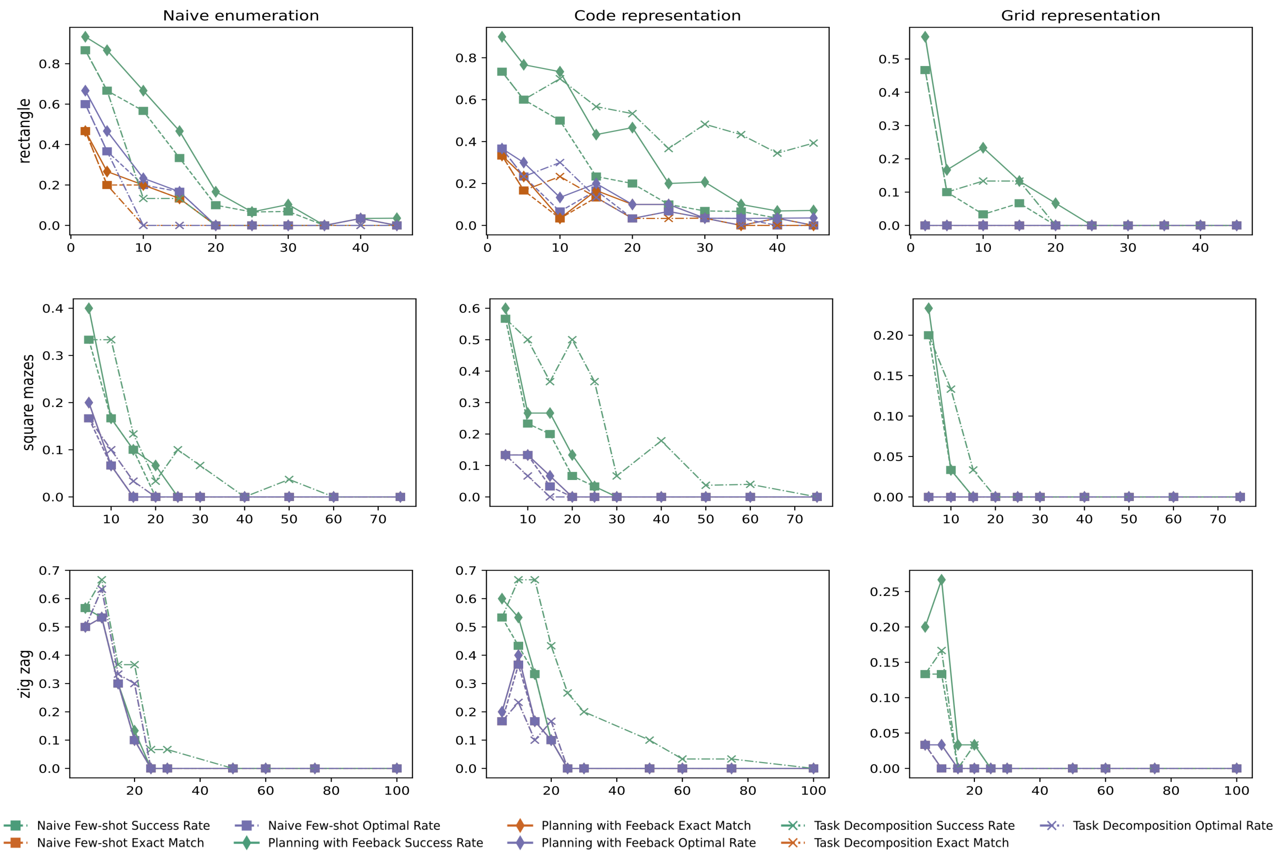


Figure 3: Path planning performance (y-axis) achieved using different prompt methodologies as a function of the ground-truth path length (x-axis). Experiments were conducted in 25×25 rectangular blocks (first row), rectangular mazes (second row), and zig-zag mazes (third row), respectively. The performance using different task representations is highlighted from left to right as a) *Naive enumeration*, b) *Code representation*, and c) *Grid representation*.

certain direction only to return and take a different route. For instance, for a case where the correct path is "*right up*", the model predicts "*down down down up up up up right*". This pattern may be because the model fails to identify the placements of entrances and obstacles on the grids.

GPT-4 is *not* mimicking the patterns in the few-shot demonstrations: The paths produced by the model differ greatly from the ground-truth plans, as can be seen in the discrepancy between the success rate and exact match scores. This indicates that the strategy adopted by the model is not the same as the one portrayed in the few-shot demonstrations (A^* search). This highlights the complexity of leveraging in-context learning in tasks that require algorithmic problem-solving and spatio-temporal reasoning. As this is an optimization problem, the algorithmic pattern may not be easily extracted from the few-shot demonstrations. Prompting with reasoning patterns that trace the algorithm (e.g., CoT (Wei et al. 2022)) may improve in this regard in short-term planning settings. However, as the number of steps increases this approach becomes inaccurate and inefficient. Our preliminary experiments show that the LLM fails to localize itself correctly on the grid and generates inaccurate reasoning chains when using CoT for our long-range planning problems.

Ablations and Error Analysis

Planning in Smaller Grids We look into whether the grid size plays a role in the ability of the LLM in path planning. Accordingly, we follow a process similar to the one used in Sec. to generate 15x15 zig zag environments. We use 2, 5, 10, 15 and 20 as in-distribution values, while out-of-distribution length generalization is evaluated using values 25, 30, 40, 50 and 60. We then run naive few-shot prompting with all three task representations. Results are showcased in Fig. 4. We notice relative improvements across all representations, particularly on short-term planning scenarios, indicating that LLMs are better at planning over shorter horizons and more simplistic environments. We observed enhancements in naive enumeration, likely because this scenario involves listing fewer obstacles.

Distance to Goal Scores To assess cases of failure, we analyze the performance of Task Decomposition. We introduce an additional metric, ***Distance to Goal***, defined as the average number of actions needed for the LLM agent to move

Table 2: Average number of input tokens needed to provide the task specifications for each representation

Task Representation	***Rect. Blocks***	***Square Mazes***	***Zig Zag***
Naive enumeration	13,734	21,360	18,063
Code representation	1,964	4,336	2,316
Grid representation	3,365	3,410	3,402

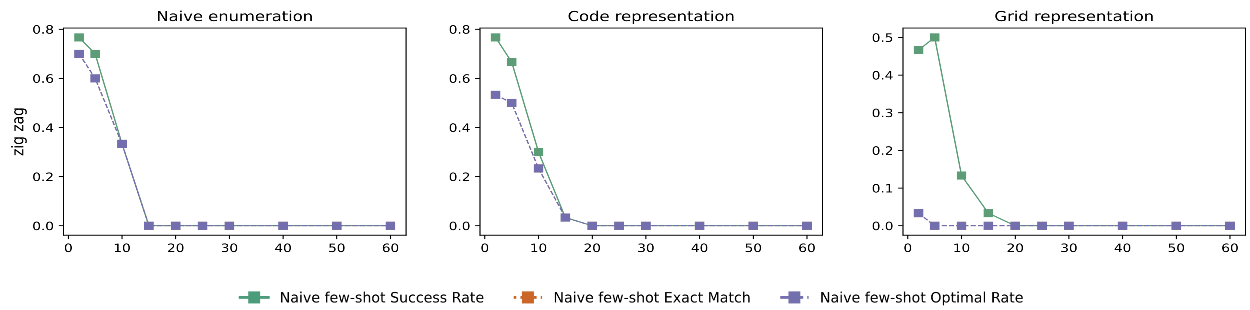


Figure 4: Path planning performance achieved on 15×15 *zig-zag mazes* environments. In cases where only the optimal rate is shown, the exact match and optimal rate values are identical.

Table 3: Average distance to goal for incorrect instances (IID/OOD).

Task Representation	***Rect. Blocks***	***Square Mazes***	***Zig Zag***
Naive enumeration	5.98/7.02	5.89/6.87	6.98/8.29
Code representation	7.09/7.03	5.57/6.59	7.43/8.40
Grid representation	8.94/8.87	6.83/7.12	8.03/8.92

from its *last valid position* to the goal location for each sub-task, calculated using the A^* algorithm. We compute the distance to goal scores on the instances that are not solved by Task Decomposition and report an average over the number of sub-tasks. The results are presented in Table 3.

GPT-4 often fails to lead the agent in the right direction: The average distances in failed cases exceed the maximum initial distance of 5. This points to the fact that GPT-4 tends to lead the agent to positions further away from the goal. This may also be a consequence of the model's inability to plan optimally.

Planning using the grid representation leads to more serious failures: The distances using the grid representation are significantly higher, significantly exceeding the maximum initial distance of 5. This further suggests that this representation is not understandable to the model.

Failures in zig-zag mazes are more significant: We notice that the distance to goal scores in zig-zag mazes are higher across all representations. This further highlights LLMs struggle to deal with this type of environment, and GPT-4's inability to produce paths that require making frequent turns. This, in turn, suggests that GPT-4 fails to perform any advanced level spatial planning/reasoning.

CONCLUSIONS

In this paper, we evaluate the ability of GPT-4 to plan through the lens of "path planning" tasks in complex geometric settings, using a variety of task representations. Our findings highlight the potential of leveraging code to provide the environment description. Decomposing a planning problem into multiple short-term planning subtasks yields promising performance. Nevertheless, performance remains subpar on long-range planning and the LLM failed to provide the optimal path in the vast majority of instances; highlighting key limitations in LLMs capability for plan generation. Addressing these issues by integrating specialized path-planning algorithms within an LLM framework can open the door to many applications in robotics and beyond.

ACKNOWLEDGEMENTS

This project was funded by George Mason Computer Science and the College of Engineering and Computing and was supported by resources provided by the Office of Research Computing at George Mason University (URL https://orc.gmu.edu) and funded in part by grants from the National Science Foundation (Awards Number 1625039 and 2018631). The authors acknowledge the use of AI language tools for light editing of the original text.

References

Aghzal, M.; Plaku, E.; and Yao, Z. 2023. Can Large Language Models be Good Path Planners? A Benchmark and Investigation on Spatial-temporal Reasoning. arXiv:2310.03249.

Ahn, M.; Brohan, A.; Brown, N.; Chebotar, Y.; Cortes, O.; David, B.; Finn, C.; Fu, C.; Gopalakrishnan, K.; Hausman, K.; Herzog, A.; Ho, D.; Hsu, J.; Ibarz, J.; Ichter, B.; Irpan, A.; Jang, E.; Ruano, R. J.; Jeffrey, K.; Jesmonth, S.; Joshi, N. J.; Julian, R.; Kalashnikov, D.; Kuang, Y.; Lee, K.-H.; Levine, S.; Lu, Y.; Luu, L.; Parada, C.; Pastor, P.; Quiambao, J.; Rao, K.; Rettinghouse, J.; Reyes, D.; Sermanet, P.; Sievers, N.; Tan, C.; Toshev, A.; Vanhoucke, V.; Xia, F.; Xiao, T.; Xu, P.; Xu, S.; Yan, M.; and Zeng, A. 2022. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. *arXiv:2204.01691*.

Anderson, P.; Wu, Q.; Teney, D.; Bruce, J.; Johnson, M.; Sünderhauf, N.; Reid, I. D.; Gould, S.; and van den Hengel, A. 2018. Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments. *2018 IEEE CVPR*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, 1877–1901. Curran Associates, Inc.

Chen, J.; Yuan, S.; Ye, R.; Majumder, B. P.; and Richardson, K. 2023a. Put Your Money Where Your Mouth Is: Evaluating Strategic Planning and Execution of LLM Agents in an Auction Arena. *arXiv:2310.05746*.

Chen, W.; Koenig, S.; and Dilkina, B. N. 2024. Why Solving Multi-agent Path Finding with Large Language Model has not Succeeded Yet. *ArXiv:2401.03630*.

Chen, W.; Ma, X.; Wang, X.; and Cohen, W. W. 2023b. Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks. *TMLR*.

Chen, Y.; Arkin, J.; Dawson, C.; Zhang, Y.; Roy, N.; and Fan, C. 2023c. AutoTAMP: Autoregressive Task and Motion Planning with LLMs as Translators and Checkers. arXiv:2306.06531.

Chevalier-Boisvert, M.; Bahdanau, D.; Lahlou, S.; Willems, L.; Saharia, C.; Nguyen, T. H.; and Bengio, Y. 2019. BabyAI: First Steps Towards Grounded Language Learning With a Human In the Loop. In *ICLR*.

Côté, M.-A.; Ákos Kádár; Yuan, X.; Kybartas, B. A.; Barnes, T.; Fine, E.; Moore, J.; Hausknecht, M. J.; Asri, L. E.; Adada, M.; Tay, W.; and Trischler, A. 2018. TextWorld: A Learning Environment for Text-based Games. *ArXiv:1806.11532*.

Ding, Y.; Zhang, X.; Paxton, C.; and Zhang, S. 2023. Task and Motion Planning with Large Language Models for Object Rearrangement. *arXiv:2303.06247*.

Gordon, D.; Kembhavi, A.; Rastegari, M.; Redmon, J.; Fox, D.; and Farhadi, A. 2018. IQA: Visual Question Answering in Interactive Environments. In *IEEE CVPR*.

Hao, S.; Gu, Y.; Ma, H.; Hong, J. J.; Wang, Z.; Wang, D. Z.; and Hu, Z. 2023. Reasoning with Language Model is Planning with World Model. In *EMNLP*.

Hart, P.; Nilsson, N.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2): 100–107.

Huang, W.; Abbeel, P.; Pathak, D.; and Mordatch, I. 2022a. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *ICML*, 9118–9147. PMLR.

Huang, W.; Xia, F.; Xiao, T.; Chan, H.; Liang, J.; Florence, P.; Zeng, A.; Tompson, J.; Mordatch, I.; Chebotar, Y.; Sermanet, P.; Brown, N.; Jackson, T.; Luu, L.; Levine, S.; Hausman, K.; and Ichter, B. 2022b. Inner Monologue: Embodied Reasoning through Planning with Language Models. arXiv:2207.05608.

Kambhampati, S.; Valmeekam, K.; Guan, L.; Stechly, K.; Verma, M.; Bhambri, S.; Saldyt, L.; and Murthy, A. 2024. LLMs Can't Plan, But Can Help Planning in LLM-Modulo Frameworks. arXiv:2402.01817.

Khot, T.; Trivedi, H.; Finlayson, M.; Fu, Y.; Richardson, K.; Clark, P.; and Sabharwal, A. 2023. Decomposed Prompting: A Modular Approach for Solving Complex Tasks. In *ICLR*.

Lin, B. Y.; Huang, C.; Liu, Q.; Gu, W.; Sommerer, S.; and Ren, X. 2023. On grounded planning for embodied tasks with language models. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press. ISBN 978-1-57735-880-0.

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

Madaan, A.; Zhou, S.; Alon, U.; Yang, Y.; and Neubig, G. 2022. Language Models of Code are Few-Shot Commonsense Learners. arXiv:2210.07128.

Mitchell, M.; Palmarini, A. B.; and Moskvichev, A. 2023. Comparing Humans, GPT-4, and GPT-4V On Abstraction and Reasoning Tasks. *ArXiv:2311.09247*.

Mueller, A.; Webson, A.; Petty, J.; and Linzen, T. 2023. In-context Learning Generalizes, But Not Always Robustly: The Case of Syntax. *arXiv:2311.07811*.

OpenAI; :; Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; Avila, R.; Babuschkin, I.; Balaji, S.; Balcom, V.; Baltescu, P.; Bao, H.; Bavarian, M.; Belgum, J.; Bello, I.; Berdine, J.; Bernadett-Shapiro, G.; Berner, C.; Bogdonoff, L.; Boiko, O.; Boyd, M.; Brakman, A.-L.; Brockman, G.; Brooks, T.; Brundage, M.; Button, K.; Cai, T.; Campbell, R.; Cann, A.; Carey, B.; Carlson, C.; Carmichael, R.; Chan, B.; Chang, C.; Chantzis, F.; Chen, D.; Chen, S.; Chen, R.; Chen, J.; Chen, M.; Chess, B.; Cho, C.; Chu, C.; Chung, H. W.; Cummings, D.; Currier, J.; Dai, Y.; Decareaux, C.; Degry, T.; Deutsch, N.; Deville, D.; Dhar, A.; Dohan, D.; Dowling, S.; Dunning, S.; Ecoffet, A.; Eleti, A.; Eloundou, T.; Farhi, D.; Fedus, L.; Felix, N.; Fishman, S. P.;

Forte, J.; Fulford, I.; Gao, L.; Georges, E.; Gibson, C.; Goel, V.; Gogineni, T.; Goh, G.; Gontijo-Lopes, R.; Gordon, J.; Grafstein, M.; Gray, S.; Greene, R.; Gross, J.; Gu, S. S.; Guo, Y.; Hallacy, C.; Han, J.; Harris, J.; He, Y.; Heaton, M.; Heidecke, J.; Hesse, C.; Hickey, A.; Hickey, W.; Hoeschele, P.; Houghton, B.; Hsu, K.; Hu, S.; Hu, X.; Huizinga, J.; Jain, S.; Jain, S.; Jang, J.; Jiang, A.; Jiang, R.; Jin, H.; Jin, D.; Jomoto, S.; Jonn, B.; Jun, H.; Kaftan, T.; Łukasz Kaiser; Kamali, A.; Kanitscheider, I.; Keskar, N. S.; Khan, T.; Kilpatrick, L.; Kim, J. W.; Kim, C.; Kim, Y.; Kirchner, H.; Kiros, J.; Knight, M.; Kokotajlo, D.; Łukasz Kondraciuk; Kondrich, A.; Konstantinidis, A.; Kosic, K.; Krueger, G.; Kuo, V.; Lampe, M.; Lan, I.; Lee, T.; Leike, J.; Leung, J.; Levy, D.; Li, C. M.; Lim, R.; Lin, M.; Lin, S.; Litwin, M.; Lopez, T.; Lowe, R.; Lue, P.; Makanju, A.; Malfacini, K.; Manning, S.; Markov, T.; Markovski, Y.; Martin, B.; Mayer, K.; Mayne, A.; McGrew, B.; McKinney, S. M.; McLeavey, C.; McMillan, P.; McNeil, J.; Medina, D.; Mehta, A.; Menick, J.; Metz, L.; Mishchenko, A.; Mishkin, P.; Monaco, V.; Morikawa, E.; Mossing, D.; Mu, T.; Murati, M.; Murk, O.; Mély, D.; Nair, A.; Nakano, R.; Nayak, R.; Neelakantan, A.; Ngo, R.; Noh, H.; Ouyang, L.; O'Keefe, C.; Pachocki, J.; Paino, A.; Palermo, J.; Pantuliano, A.; Parascandolo, G.; Parish, J.; Parparita, E.; Passos, A.; Pavlov, M.; Peng, A.; Perelman, A.; de Avila Belbute Peres, F.; Petrov, M.; de Oliveira Pinto, H. P.; Michael; Pokorny; Pokrass, M.; Pong, V.; Powell, T.; Power, A.; Power, B.; Proehl, E.; Puri, R.; Radford, A.; Rae, J.; Ramesh, A.; Raymond, C.; Real, F.; Rimbach, K.; Ross, C.; Rotsted, B.; Roussez, H.; Ryder, N.; Saltarelli, M.; Sanders, T.; Santurkar, S.; Sastry, G.; Schmidt, H.; Schnurr, D.; Schulman, J.; Selsam, D.; Sheppard, K.; Sherbakov, T.; Shieh, J.; Shoker, S.; Shyam, P.; Sidor, S.; Sigler, E.; Simens, M.; Sitkin, J.; Slama, K.; Sohl, I.; Sokolowsky, B.; Song, Y.; Staudacher, N.; Such, F. P.; Summers, N.; Sutskever, I.; Tang, J.; Tezak, N.; Thompson, M.; Tillet, P.; Tootoonchian, A.; Tseng, E.; Tuggle, P.; Turley, N.; Tworek, J.; Uribe, J. F. C.; Vallone, A.; Vijayvergiya, A.; Voss, C.; Wainwright, C.; Wang, J. J.; Wang, A.; Wang, B.; Ward, J.; Wei, J.; Weinmann, C.; Welihinda, A.; Welinder, P.; Weng, J.; Weng, L.; Wiethoff, M.; Willner, D.; Winter, C.; Wolrich, S.; Wong, H.; Workman, L.; Wu, S.; Wu, J.; Wu, M.; Xiao, K.; Xu, T.; Yoo, S.; Yu, K.; Yuan, Q.; Zaremba, W.; Zellers, R.; Zhang, C.; Zhang, M.; Zhao, S.; Zheng, T.; Zhuang, J.; Zhuk, W.; and Zoph, B. 2023. GPT-4 Technical Report.

OpenAI. 2023. GPT-4V(ision) System Card.

Pallagani, V.; Muppasani, B.; Murugesan, K.; Rossi, F.; Srivastava, B.; Horesh, L.; Fabiano, F.; and Loreggia, A. 2023. Understanding the Capabilities of Large Language Models for Automated Planning. *ArXiv*, abs/2305.16151.

Prasad, A.; Koller, A.; Hartmann, M.; Clark, P.; Sabharwal, A.; Bansal, M.; and Khot, T. 2023. ADaPT: As-Needed Decomposition and Planning with Language Models. *arXiv*.

Puerto, H.; Tutek, M.; Aditya, S.; Zhu, X.; and Gurevych, I. 2024. Code Prompting Elicits Conditional Reasoning Abilities in Text+Code LLMs. *arXiv:2401.10065*.

Qiu, L.; Hu, H.; Zhang, B.; Shaw, P.; and Sha, F. 2021. Systematic Generalization on gSCAN: What is Nearly Solved and What is Next? *arXiv:2109.12243*.

Raman, S. S.; Cohen, V.; Rosen, E.; Idrees, I.; Paulius, D.; and Tellex, S. 2022. Planning With Large Language Models Via Corrective Re-Prompting. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Ruis, L.; Andreas, J.; Baroni, M.; Bouchacourt, D.; and Lake, B. M. 2020. A Benchmark for Systematic Generalization in Grounded Language Understanding. In *Advances in Neural Information Processing Systems*, volume 33, 19861–19872.

Shridhar, M.; Thomason, J.; Gordon, D.; Bisk, Y.; Han, W.; Mottaghi, R.; Zettlemoyer, L.; and Fox, D. 2020. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *IEEE CVPR*.

Shridhar, M.; Yuan, X.; Côté, M.-A.; Bisk, Y.; Trischler, A.; and Hausknecht, M. 2021. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *Proceedings of ICLR*.

Silver, T.; Dan, S.; Srinivas, K.; Tenenbaum, J. B.; Kaelbling, L. P.; and Katz, M. 2023. Generalized Planning in PDDL Domains with Pretrained Large Language Models. *arXiv preprint arXiv:2305.11014*.

Song, C. H.; Wu, J.; Washington, C.; Sadler, B. M.; Chao, W.-L.; and Su, Y. 2023. LLM-Planner: Few-Shot Grounded Planning for Embodied Agents with Large Language Models. arXiv:2212.04088.

Sun, H.; Zhuang, Y.; Kong, L.; Dai, B.; and Zhang, C. 2023. AdaPlanner: Adaptive Planning from Feedback with Language Models. *arxiv:2305.16653*.

Tong, S.; Liu, Z.; Zhai, Y.; Ma, Y.; LeCun, Y.; and Xie, S. 2024. Eyes Wide Shut? Exploring the Visual Shortcomings of Multimodal LLMs. *arXiv:2401.06209*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; Bikel, D.; Blecher, L.; Ferrer, C. C.; Chen, M.; Cucurull, G.; Esiobu, D.; Fernandes, J.; Fu, J.; Fu, W.; Fuller, B.; Gao, C.; Goswami, V.; Goyal, N.; Hartshorn, A.; Hosseini, S.; Hou, R.; Inan, H.; Kardas, M.; Kerkez, V.; Khabsa, M.; Kloumann, I.; Korenev, A.; Koura, P. S.; Lachaux, M.-A.; Lavril, T.; Lee, J.; Liskovich, D.; Lu, Y.; Mao, Y.; Martinet, X.; Mihaylov, T.; Mishra, P.; Molybog, I.; Nie, Y.; Poulton, A.; Reizenstein, J.; Rungta, R.; Saladi, K.; Schelten, A.; Silva, R.; Smith, E. M.; Subramanian, R.; Tan, X. E.; Tang, B.; Taylor, R.; Williams, A.; Kuan, J. X.; Xu, P.; Yan, Z.; Zarov, I.; Zhang, Y.; Fan, A.; Kambadur, M.; Narang, S.; Rodriguez, A.; Stojnic, R.; Edunov, S.; and Scialom, T. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models.

Valmeekam, K.; Marquez, M.; Olmo, A.; Sreedharan, S.; and Kambhampati, S. 2023a. PlanBench: An Extensible Benchmark for Evaluating Large Language Models on Planning and Reasoning about Change. In *NeurIPS Datasets and Benchmarks Track*.

Valmeekam, K.; Sreedharan, S.; Marquez, M.; Olmo, A.; and Kambhampati, S. 2023b. On the Planning Abilities of Large Language Models (A Critical Investigation with a Proposed Benchmark). *arXiv:2302.06706*.

Wang, Z.; Cai, S.; Chen, G.; Liu, A.; Ma, X.; and Liang, Y. 2023. Describe, Explain, Plan and Select: Interactive

Planning with Large Language Models Enables Open-World Multi-Task Agents. *arXiv:2302.01560*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; brian ichter; Xia, F.; Chi, E. H.; Le, Q. V.; and Zhou, D. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. In Oh, A. H.; Agarwal, A.; Belgrave, D.; and Cho, K., eds., *Advances in Neural Information Processing Systems*.

Xiao, H.; and Wang, P. 2023. LLM A*: Human in the Loop Large Language Models Enabled A* Search for Robotics. arXiv:2312.01797.

Xie, Y.; Yu, C.; Zhu, T.; Bai, J.; Gong, Z.; and Soh, H. 2023. Translating Natural Language to Planning Goals with Large-Language Models. *arXiv:2302.05128*.

Yang, Y.; and Tomar, A. 2023. On the Planning, Search, and Memorization Capabilities of Large Language Models. *arXiv:2309.01868*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. R. 2023a. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In *NeurIPS*.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K. R.; and Cao, Y. 2023b. ReAct: Synergizing Reasoning and Acting in Language Models. In *ICLR*.

Yue, X.; Ni, Y.; Zhang, K.; Zheng, T.; Liu, R.; Zhang, G.; Stevens, S.; Jiang, D.; Ren, W.; Sun, Y.; Wei, C.; Yu, B.; Yuan, R.; Sun, R.; Yin, M.; Zheng, B.; Yang, Z.; Liu, Y.; Huang, W.; Sun, H.; Su, Y.; and Chen, W. 2023. MMMU: A Massive Multi-discipline Multimodal Understanding and Reasoning Benchmark for Expert AGI. *arXiv:2311.16502*.

Zhao, Z.; Lee, W. S.; and Hsu, D. 2023. Large Language Models as Commonsense Knowledge for Large-Scale Task Planning. In *RSS 2023 Workshop on Learning for Task and Motion Planning*.

Zhou, D.; Schärli, N.; Hou, L.; Wei, J.; Scales, N.; Wang, X.; Schuurmans, D.; Cui, C.; Bousquet, O.; Le, Q. V.; and Chi, E. H. 2023. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. In *ICLR*.