Analyzing Capabilities of GPT-3 for Intelligent Task Execution with ALFRED

Zain-ul-Abideen Nasir
Computer Science Department
SUNY Binghamton
NY, USA
znasir1@binghamton.edu

Aniruddha Rakshit
Computer Science Department
SUNY Binghamton
NY, USA
arakshi1@binghamton.edu

***Abstract*—Large language models have shown remarkable success in various natural language processing tasks, but their application in interpreting grounded instructions for everyday tasks is relatively new. Recent techniques involve mapping instructions to simpler tasks that can be executed in a static environment. Such techniques rely on the simplicity of initial instructions. This paper explores the application of large language model like GPT-3 in interpreting grounded instructions for everyday tasks through the use of the ALFRED benchmark. The ALFRED benchmark presents a set of tasks that require agents to perform everyday activities, such as making coffee or setting the table, based on natural language instructions and visual observations. We demonstrate how large language models can be leveraged to improve performance on the ALFRED benchmark by fine-tuning them on the task-specific training data. We discuss the implications of our findings for the development of more robust language models for interpreting grounded instructions in real-world scenarios. This project introduces a technique to incorporate GPT-3 with ALFRED in order to map more complex instructions to individual sub-tasks. The experiments are conducted in iThor [1] and RoboThor [2] environments with everyday tasks of varying complexity. The contributions of the project are going to be: 1) to demonstrate GPT-3's capabilities in breaking up everyday tasks into executable actions and 2) to analyze the extent to which the actions can be executed in static environments.**

***Index Terms*—Task execution, natural language instructions, task success, goal success**

I. Introduction

Everyday task execution through natural language instructions is an important field in robotics as it makes it more intuitive to utilize agents in normal day-to-day operations. However, they also face certain challenges due to the complexity of natural language. For example, simple instructions such as "Clean the dining table" can have different meanings, depending on the context and the state of the environment. In order to extend the possibilities that an agent can handle, it is important to incorporate preliminary instruction processing techniques.

In our project, we take a look at how GPT-3 [3] can be utilized for instruction processing for task execution using ALFRED bench marking [4]. GPT-3 is a large language model that is suitable for prompt completion. It is capable of breaking down complex instructions into simpler ones, given the state of the environment and agent's capabilities. [5] The simpler instructions can then be provided to ALFRED to generate a sequence of actions that a robotic agent can then execute in simulation environments, including iThor [1] and RoboThor [2].

II. Proposal

A. Project Description

The goal of this project is to extend ALFRED's natural language instruction mapping system by incorporating GPT-3 [3] into the pipeline. As an auto-regressive language model, GPT-3 [3] has the capability to produce text in continuation of a given prompt. In the context of our project, GPT-3 [3] can be used to break down complex everyday tasks into simpler sub-instructions. ALFRED, a benchmark for learning a mapping from instructions to a sequence of actions, can then take the sub-instructions as inputs and output a set of actions that can then be executed in a static environment. The combination of GPT-3 [3] and ALFRED creates a 2-step pipeline that starts with a complex task and ends with a sequence of actions fulfilling that task.

The first part of our project involves prompt engineering for GPT-3. Currently, GPT-3 is capable of breaking down a single task into multiple simpler tasks. However, to ensure that the simpler tasks can be executed in a static environment, GPT-3 would require domain knowledge from the environment. Through careful prompt engineering and parameter optimization, it can be ensured the GPT-3 only outputs tasks that can be executed in the given environment.

The second part of our project involves combining GPT-3 with ALFRED and experimenting in simulated static environments using iTHOR [1] and RoboTHOR [2]. GPT-3's ability to break a task into sub-tasks depends on the complexity of the environment available to the robot. For example, in a static kitchen environment, given the task of making coffee in a coffee maker, GPT-3 can easily break this task into sub-tasks starting from opening up the coffee maker and putting in ingredients and ending with pouring brewed coffee into the cup. However, the accuracy of these sub-tasks can only be tested in a kitchen environment which includes all the apparatus required, including an interactive coffee maker. This project aims to experiment with iThor [1] and RoboThor

[2] environments to analyze the execution of the sub-tasks generated by GPT-3 and ALFRED.

B. Importance and Application

Natural language is a rich, intuitive mechanism by which humans can interact with systems around them, offering sufficient signals to support robot task planning. Task execution through natural language instruction eliminates the need for a detailed map of the environment, equipping the robot with a certain level of autonomy. The key challenge in this is that human instructions are complex combinations of words and are highly depended on contextual information. A robot must be able to translate the context to its environment and break down the instructions into a sequence of actions that each lead to a specific goal. By exploring the current industry standards in natural language processing and task execution, GPT-3 and ALFRED, we will be able to analyze the utility of such systems and identify points of improvement.

Through our project, we will be able to understand the extent to which Large Language Models (LLMs) can be used in robotic task execution. LLM-based text generators are not specifically trained for task execution. Their training datasets are vast, which results in a wide variety of possible outputs. For robotic task execution, this leads to a lower predictability which is undesirable. Our project aims to explore how predictability can be improved through prompt engineering so that LLMs map instructions to tasks while keeping the restrictions of a static environment in consideration.

C. Limitations of ALFRED

ALFRED is a benchmark for interpreting grounded instructions for everyday tasks. It is a popular benchmark used in the field of natural language processing (NLP) and robotics. While ALFRED has several strengths, it also has some limitations:

Limited Scope: ALFRED is limited to a specific set of household tasks, such as setting tables, cleaning kitchens, and putting away groceries. Therefore, the benchmark may not be generalizable to other environments or tasks.

Simplified Environments: ALFRED's environments are simplified versions of real-world environments. They lack the complexity and variability of real-world environments, which could limit the generalizability of the benchmark.

Lack of Interactivity: ALFRED's environments do not provide a high degree of interactivity with the user. The user cannot ask questions or provide feedback to the agent, which could limit the agent's ability to learn and adapt to new situations.

Limited Evaluation Metrics: ALFRED's evaluation metrics are primarily focused on task completion and do not take into account other important aspects of natural language understanding, such as semantic similarity or pragmatic appropriateness.

Limited Interaction with Humans: It's environments do not provide much opportunity for the robotic agent to interact with humans, which could limit the agent's ability to learn from and adapt to human behaviors and preferences.

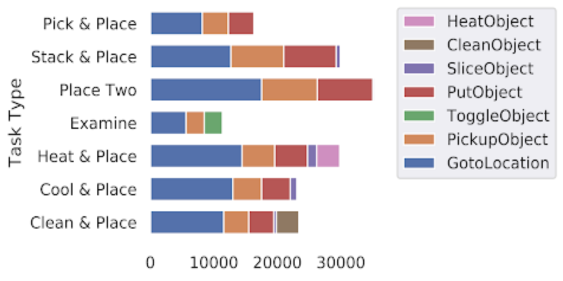


Fig. 1. Limitations of ALFRED

Limited Generalization: The tasks in ALFRED are designed to be solved using specific actions and objects, which may not generalize to new or unseen actions or objects. This could limit the agent's ability to adapt to novel situations.

Simplified Language: The language used in ALFRED's instructions is simplified compared to natural language, which may not reflect the complexity and variability of natural language expressions used in real-world environments.

Lack of Uncertainty: It's environments do not include uncertainty, which is a common feature of real-world environments. This could limit the agent's ability to deal with unexpected situations or errors. It cannot deal with dynamic environments like busy roads where vehicles are running at a high speed or any playing ground.

Moreover, a sequence-to-sequence model with progress monitoring is evaluated using ALFRED after being successful in other vision and language-based navigation challenges as shown in Fig. 1. The total task success rates are low, despite the fact that this model is reasonably adept at achieving some sub goals, for instance operating microwaves is consistent across heat and place tasks. With sub problems like visual semantic navigation, object detection, referring expression grounding and action grounding, the long horizon of ALFRED tasks presents a significant challenges.

Overall, while ALFRED is a useful benchmark for evaluating the performance of natural language understanding and robotic agents, it has limitations that need to be considered when interpreting its results.

D. Combining GPT-3 and ALFRED

GPT-3 is a state-of-the-art language model that can generate natural language text. This text contains the goal for a specific environment. One way to combine GPT-3 and ALFRED is to use GPT-3 to generate natural language instructions for ALFRED's tasks, as shown in Fig. 2. We will follow the below steps to combine the GPT-3 and ALFRED

Instruction Generation: GPT-3 can be used to generate natural language instructions which contains a goal for the task, which can be used to communicate the task to the robotic agent. The agent can then use its perception and action capabilities to complete the task. Task Generation: ALFRED's tasks can be randomly generated by a script or tool, specifying the goal of the task and the environment in which it takes

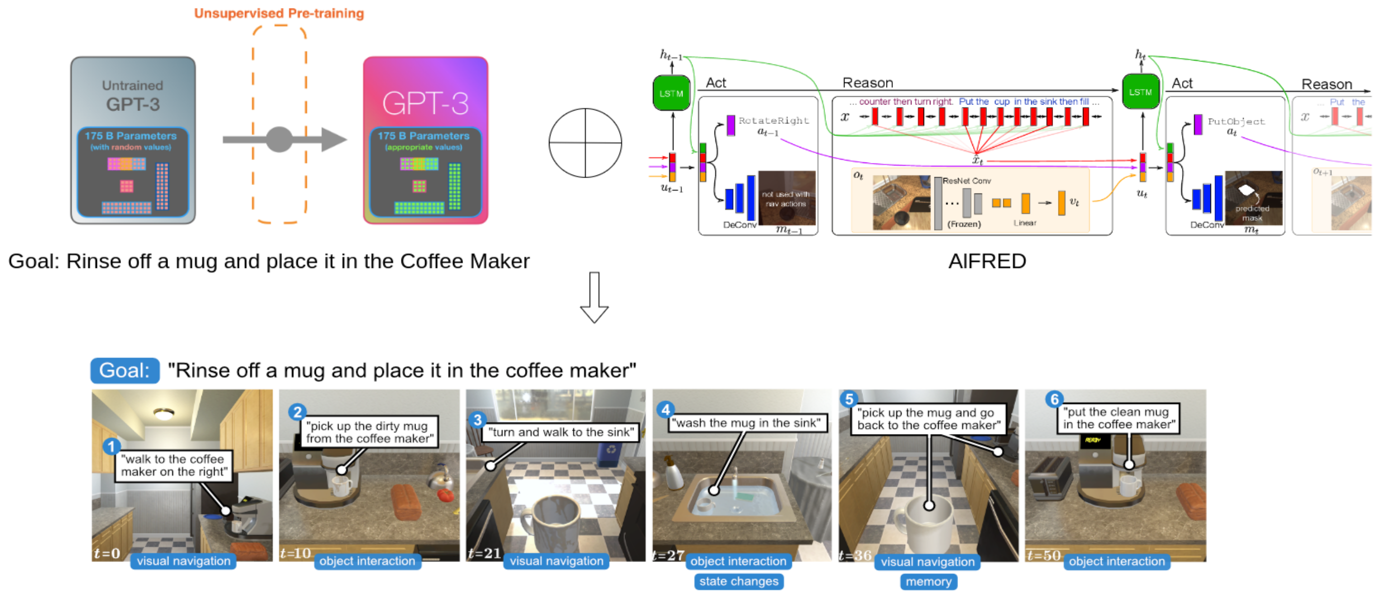

Unsupervised Pre-training
Untrained GPT-3
175 B Parameters (with random values)
GPT-3
175 B Parameters (appropriate values)
Goal: Rinse off a mug and place it in the Coffee Maker
Act
Reason
LSTM
RotateRight
PutObject
not used with nav actions
predicted mask
ResNet Conv
(Frozen)
Linear
DeConv
AlFRED
Goal: "Rinse off a mug and place it in the coffee maker"
"walk to the coffee maker on the right"
"pick up the dirty mug from the coffee maker"
"turn and walk to the sink"
"wash the mug in the sink"
"pick up the mug and go back to the coffee maker"
"put the clean mug in the coffee maker"
visual navigation
object interaction
state changes
memory

Fig. 2. Our Approach

place. For example, a task could be to make a cup of tea in the kitchen.

Agent Execution: The robotic agent can use its perception capabilities to understand the closed environment and its action capabilities to interact with the environment to complete the task. The agent can also use its language processing capabilities to interpret the instructions generated by GPT-3.

Evaluation: The agent's performance can be evaluated based on how far it completes the task by achieving a goal and also based on how well it understands the natural language instructions.

In summary, we will generate a goal and pass it to GPT-3 which will be divided into subgoals by GPT-3. These subgoals will be broken down into a sequence of actions by ALFRED.Combining GPT-3 and ALFRED in this way could help to improve the natural language understanding and generation capabilities of robotic agents, making them more effective at completing everyday tasks. However, it is important to note that this approach may have some limitations, such as the difficulty of generating natural language instructions that are appropriate for a specific task or environment. Additionally, the cost of using GPT-3 may be prohibitive for some applications.

III. Research Plan

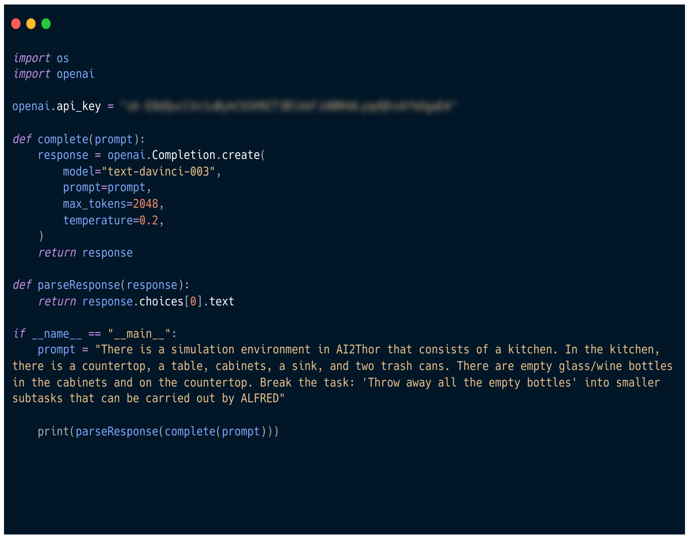
The project is divided into two major tasks: 1) GPT-3 experimentation to figure out the most optimal prompt format and parameter values, and 2) ALFRED and simulation environment setup to test GPT-3 instruction outputs. Zain will be responsible for GPT-3 experimentation while Aniruddha will be responsible for ALFRED setup.

A. *GPT-3 parameter optimization and experimentation*

For this project, we can utilize GPT-3 in two ways. The first one is to use the pre-trained models and solely rely on prompt engineering. On the other hand, we can fine-tune the pre-trained models on custom dataset. In order to cater to the time constraint, we have decided to choose the first option of using pre-trained model. This would eliminate the need for collecting hundreds of training prompts. Furthermore, it would allow us to understand the capability the minimum capability of GPT-3 that can be achieved without further fine-tuning. Relying on prompt engineering prevents our method from relying on a single model. In the longer run, if GPT-3 evolves and its models change, we would not have to retrain the newer models because our method would solely rely on a prompt.In addition to an input prompt, GPT-3 requires certain parameters which govern various features of the predicted output, including *temperature* and *top-P*.

The *temperature* parameter governs the creativity in GPT-3's responses. In the context of robotic task execution, we will have to choose between high predictability, low randomness and low predictability, high randomness. For everyday tasks, the intuition is that a robot should be taking the same steps if it prompted to complete the same task. Therefore, high predictability, low randomness would be a better option. However, to form a concrete conclusion, further experimentation is required which will be carried out as part of this project.

The *top-P* governs the diversity of the tokens used in the predicted tokens. A low *top-P* value generates responses containing only commonly used words while a high *top-P* value generates responses that also contain less common words. As part of experimentation, we plan on analyzing task execution by varying *top-P's* value to decide on the most optimal choice.



```python
import os
import openai

openai.api_key = [redacted]

def complete(prompt):
    response = openai.Completion.create(
        model="text-davinci-003",
        prompt=prompt,
        max_tokens=2048,
        temperature=0.2,
    )
    return response

def parseResponse(response):
    return response.choices[0].text

if __name__ == "__main__":
    prompt = "There is a simulation environment in AI2Thor that consists of a kitchen. In the kitchen,
there is a countertop, a table, cabinets, a sink, and two trash cans. There are empty glass/wine bottles
in the cabinets and on the countertop. Break the task: 'Throw away all the empty bottles' into smaller
subtasks that can be carried out by ALFRED"

    print(parseResponse(complete(prompt)))
```

Fig. 3. GPT3 Setup

B. *ALFRED simulation setup for testing (Aniruddha)*

We will evaluate the baseline models in the iTHOR and RoboThor simulation environments. When evaluating on test folds, we will run models with the lowest validation loss. Episodes that exceed 500 steps or cause more than 10 failed actions are terminated. We will evaluate our model based on our accuracy with respect to our iterations. Failed actions arise from bumping into walls or predicting action interaction masks for incompatible objects, such as attempting to pick up a counter top. These limitations encourage efficiency and reliability. We assess the overall and partial success of models' task executions across episodes. We will also evaluate whether task goal-conditions have been achieved for example, coffee has been made. For both of our environments we will analyze task success and goal-condition success. For instance, "Take a coffee mug" is a goal condition which relies on navigating to coffee mug and grabbing it.

IV. Intellectual Merit

Major companies like Amazon are introducing household robots that are meant to be used for various tasks, ranging from remote monitoring to house-cleaning. According to a report published by Straits Research, the global household robots market size was valued at USD 6.81 billion in 2021, and is expected to reach USD 30.7 billion by 2030, growing at a CAGR of 20.7%. Despite its rapid growth, there is still room for improvement when it comes to natural language instruction. Most house-cleaning robots, such as Roomba, are too specialized to their tasks and fail in surprise circumstances. Therefore, it is important to equip such robots with the ability to understand natural language commands and execute them in dynamic environments. This project will give us an insight into how large language models can be combined with instruction mapping techniques to incorporate natural language into task execution.

V. Implementation

A. *GPT3 Setup*

For our GPT-3 setup, we used basic customization. The model 'text-davinci-003' was used for prompt completion, which in our case was generating sub-tasks from a complex task. After prompt experimentation, which is explained in the next section, we decided to set the maximum token limit to 2048 and the temperature to 0.2. A python script was used to send prompts to the OpenAI API and receive a response. Initially, retraining the base model seemed like a feasible option. However, retraining requires extensive data which was impractical considering the scope of this project. Therefore, we eventually settled on using the base model which proved to be powerful enough for static iThor environments.

B. *Prompt Engineering*

The most important part of GPT-3 prompt completion is prompt engineering. Accuracy and results can vary broadly depending on the specificity of the prompt. During our experimentation, we decided to test two different kind of prompts. The first prompt, which proved to be more efficient and accuract, involved an elaborate explanation of the environment and objects available to the robot. One such example is: "There is a simulation environment in AI2-Thor that consists of a kitchen. In the kitchen, there is a counter-top, a table, cabinets, a sink, and two trash cans. There is an empty wine/glass bottle in the cabinet and another one on the counter top. Break the task: 'Throw away all the empty bottles' into smaller pick-and-place ALFRED sub-tasks". When using prompts like these, GPT-3 was forced to avoid drifting away from the environment, allowing it to stay within the constraints of the environment and providing sub-tasks that were possible to execute in the concerned environment. The second type of prompt, which proved to be less accurate due to the variation in responses genereted by GPT-3, did not involve any explanation of the environment. GPT-3 was allowed to guess the domain knowledge just by providing the name of the environment, such as "kitchen environment in Ai2-Thor". Such prompts proved to be less accurate because they allowed GPT-3 to guess the items available in the environment. In most cases, the subtasks generated involved movements or objects that were not available in the environment and thus could not be passed to ALFRED for further interpretation.

C. *GPT3 Parameter Optimization*

GPT-3 provides extensive control over the generated responses through parameters that you can fine tune according to your needs. For our use case, we were primarily focused on token limit, temperature, and presence penalty.

Token limit parameter decides the maximum count of prompt plus the generated text. For task generation, token limit did not have significant impact on performance, except in the extreme case where setting the limit to a very low value such as 10 would prevent the generation of tasks as the response would get cut off before it was even finished.

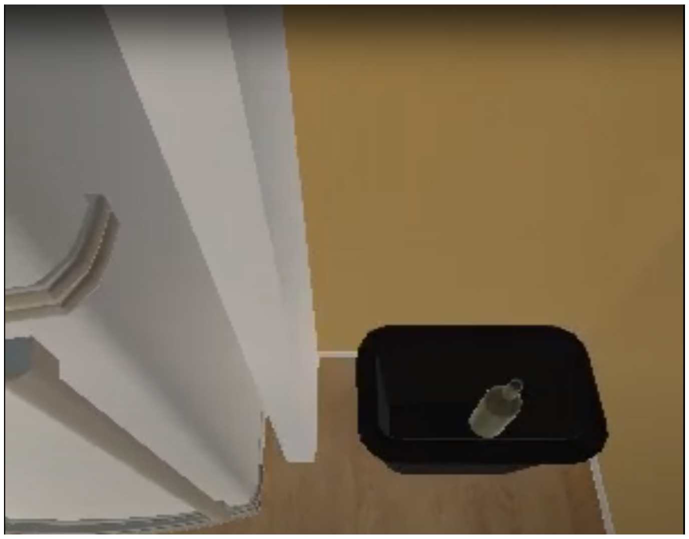


Fig. 4. Single Item Execution: Throw away all the empty bottles

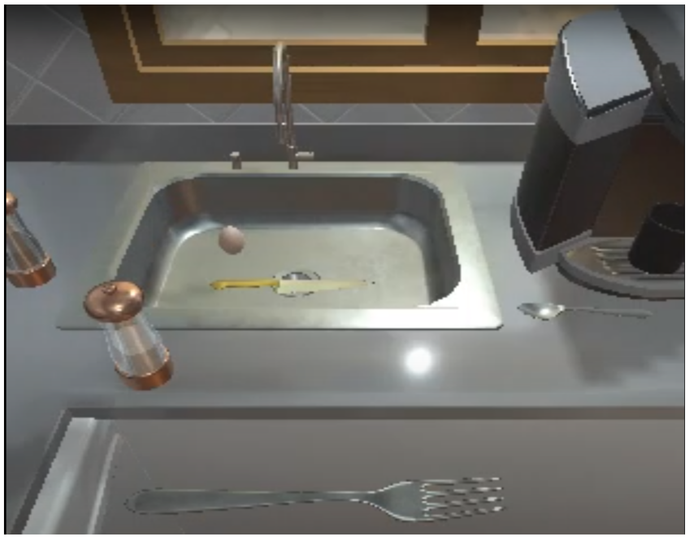


Fig. 5. Multi Item Execution: Put all the utensils in the sink

Temperature parameter decides the randomness of output. It's value ranges from 0 (deterministic) to 2 (random). After experimentation, we noticed that a high temperature value reduced success rate and repeatability because GPT-3 was always generating unique responses and deviating from the environment constraints. Setting the temperature value to 0.2 allowed us to predict the responses to a certain extent and rely on them to repeat tests in a static environment.

Presence penalty controls GPT-3's tendency to talk about new topics. It's value ranges from -2.0 (reluctant) to 2.0 (encouraged). Our experiments showed that a negative value forced GPT-3 o stay within the environment constraints and only generate sub-tasks that could be executed using the domain knowledge provided in the prompt. A negative value also increased repeatability because GPT-3 made sure its responses only used information provided in the prompt.

D. Tests and Results

We tested our project through three different types of task execution in ALFRED environment. Single item execution, multi-item execution, and successive execution.

Firstly, We tested with single item execution where we gave a prompt "Throw away all the empty bottles" to the GPT-3. Then GPT-3 divides the tasks into two smaller sub-tasks, "1. Pick up the empty glass/wine bottle from the counter top and place in the trash can. 2. Pick up the empty glass/wine bottle from the cabinet and place in the trash can." Then ALFRED environment divides the two sub-tasks into more smaller sub-tasks as shown in Fig. 4: "Step 1:Turn around and take a step, then turn right and begin walking across the room, hand a left and walk up the area of kitchen counter to the left of the stove. Step 2: Pick up the glass bottle that is on the kitchen counter to the left of the stove. Step 3: Turn right and walk around the kitchen island back to the small black trash can in the corner of the room. Step 4: Place the second glass bottle into the trash can to the left the bottle that is already in there. Step 5: Turn right and walk up to the kitchen counter, then look up at the upper cabinets. Step 6: Open the left hand door of the rightmost cupboard in front of you and remove the glass bottle, then close the door. Step 7: Turn around and begin walking across the room, stop at small black trash can just beyond the fridge and turn left to face it. Step 8: Place the glass bottle into the small black trash can on the right side".

Secondly, for multi-item execution we gave a prompt to GPT-3 as "Put all utensils in the sink" Then GPT-3 divides the task into two more sub-tasks like : "Pick up fork from counter top and pace in sink. Pick up fork from tabletop and place in sink." ALFRED makes these two tasks into more smaller sub-tasks as show in Fig 5. "Step 1: Turn right and walk over to the oven. Step 2: Pick up the metal fork that is to the right of the loaf of bread on the counter. Step 3: Turn right and walk over to the kitchen sink. Step 4: Place the metal fork into the sink basin. Step 5: Turn right and begin walking across the room, then turn right again and walk up to the end of the kitchen counter. Step 6: Pick up the metal fork that is on the kitchen counter. Step 7: Turn left and begin walking forward, then turn right and walk over to the kitchen sink. Step 8: Place the fork into the sink basin".

Finally, for successive execution, the prompt is "Collect all stationery, including the CD from table and move to shelf". In ALFRED environment, the agent will collect all the stationery like pencil, pen and CD and put all the utensils on the shelf as shown in Fig 6 . As, we mentioned earlier that, utensils consist of pencil, pen and CD, the agent will look for these three objects in the environment. Moreover, the agent will move from any location to the object location according to the prompt and complete the task successively.

VI. Limitation

First, GPT-3 is a powerful language model that can generate coherent and fluent text, but it is not specifically designed or trained for robotic task generation. This means that it may not

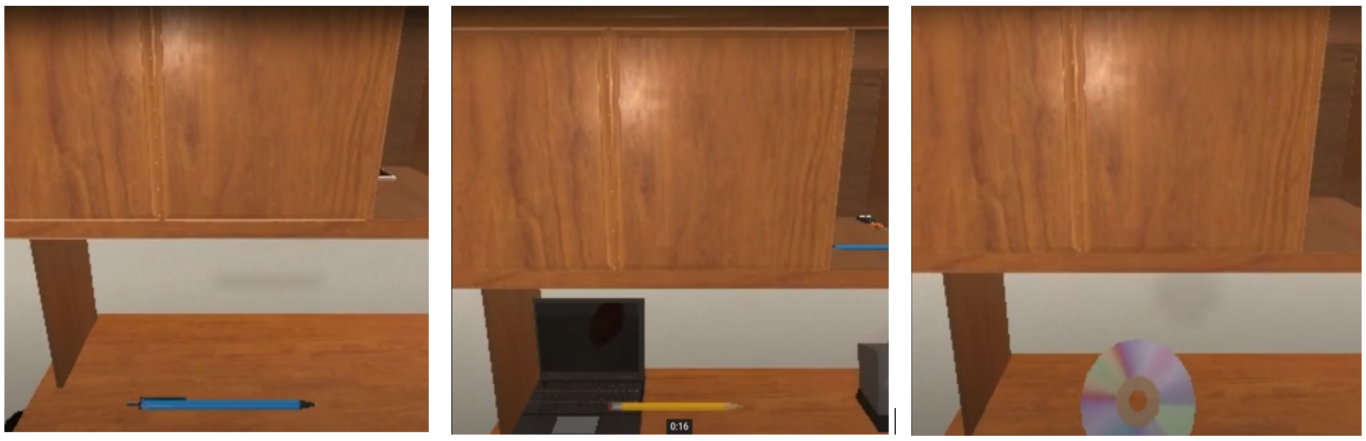


Fig. 6. Successive Execution: Collect all stationery, including the CD from table and move to shelf

always produce the most accurate or effective instructions for robots to follow in completing tasks.Because GPT-3 is a probabilistic model, it can be difficult to predict or control the output it generates. This lack of determinism can make it challenging to use in situations where precise outcomes are necessary, such as in robotics. For example: If We give a prompt GPT3, Make a coffee which is a high level prompt. The GPT3 model will ask which type of coffee flavor do we like. Is it Cappuccino, Latte, Americano, Espresso. If Cappuccino, which flavor do we like etc. Moreover, fine-tuning a language model like GPT-3 for a specific task requires a large amount of computational resources, which can be expensive and time-consuming. This can make it difficult for researchers or developers to optimize the model for specific robotic applications. In order for GPT-3 to generate accurate and effective instructions for robots, it needs to have a complete understanding of the environment in which the robot is operating. This can be a challenging task, as it requires the model to have access to a wide range of data and information about the environment.While fine-tuning GPT-3 can be computationally expensive, it is often necessary in order to optimize the model for specific applications. Testing the model with unseen datasets can help ensure that it is able to generate accurate and effective instructions in a variety of contexts. ALFRED is a benchmark dataset for robotic task planning and execution, but it has limitations in terms of the complexity of the environments it models. By extending ALFRED with other more robust environments, such as RoboThor, researchers can more accurately simulate real-world scenarios and test the effectiveness of GPT-3 and other language models for robotic applications.

VII. Conclusion

While GPT-3 is a powerful language model that can generate coherent and fluent text, it is not specifically designed or trained for robotic task generation. Fine-tuning the model for specific robotic applications can be computationally expensive and time-consuming, and the lack of determinism can make it challenging to use in situations where precise outcomes are necessary

References

[1] E. Kolve, R. Mottaghi, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, "AI2-THOR: an interactive 3d environment for visual AI," *CoRR*, vol. abs/1712.05474, 2017. [Online]. Available: http://arxiv.org/abs/1712.05474

[2] M. Deitke, W. Han, A. Herrasti, A. Kembhavi, E. Kolve, R. Mottaghi, J. Salvador, D. Schwenk, E. VanderBilt, M. Wallingford, L. Weihs, M. Yatskar, and A. Farhadi, "Robothor: An open simulation-to-real embodied ai platform," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *CoRR*, vol. abs/2005.14165, 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[4] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, "Alfred: A benchmark for interpreting grounded instructions for everyday tasks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[5] Y. Fu, H. Peng, A. Sabharwal, P. Clark, and T. Khot, "Complexity-based prompting for multi-step reasoning," 2023.