Spontaneous Response to Dynamic Environments via GPT 3.5

Richard Wu, Rohit Malhotra

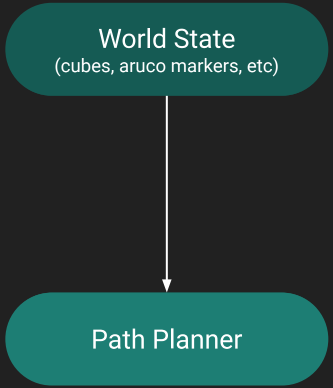
Background

Cozmo learns world state with recognizable objects: cubes, aruco markers, etc. Path planner uses the world state to create an action plan to goal state

Problems -

1) Unrecognizable objects in the way are not considered
2) Path planner precalculates if goal is reachable given world state (no goal collisions, etc)
3) Executes deterministic steps to reach goal, ignoring dynamic changes to the environment

There is rerouting/abort sequence capabilities given dynamic changes to the environment, with both recognizable and unrecognizable objects !!



Goal Overview

Make cozmo spontaneously respond to changes in environment that prevent successful execution of path plan. This would mainly come from objects that are/aren’t in the world map and are discovered to be blockers while cozmo in the midst of executing a path plan.

General Approach:

- Use GPT 3.5 as a central nervous system for cozmo
- Make high level decisions on how to assess cozmo’s environment and identify blockers
- Make ultimate decision on when to interrupt cozmo and steal control from path planner
- Trigger abort + reroute sequence

Challenges:

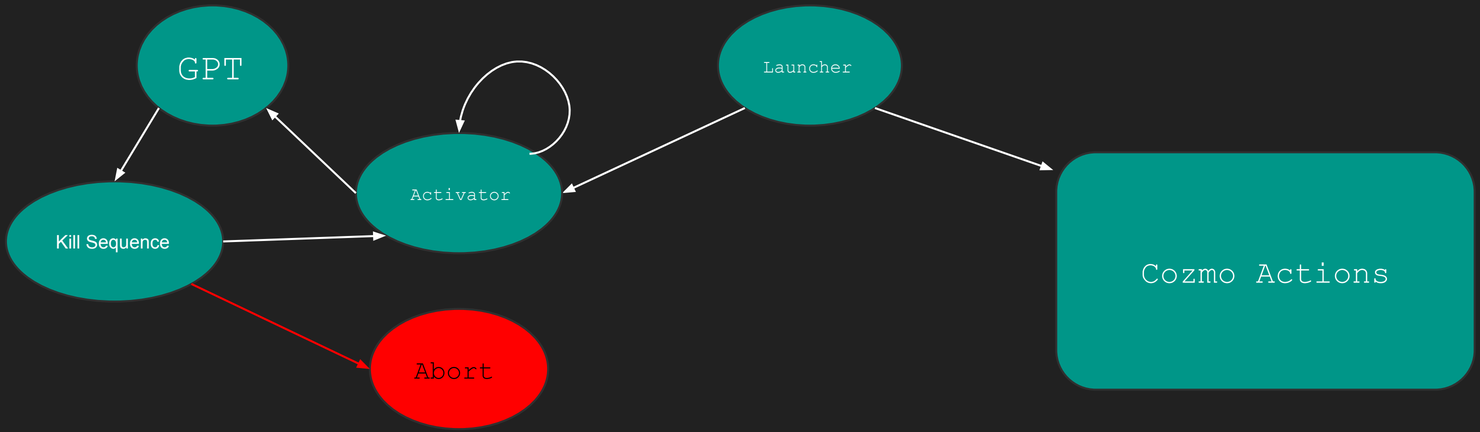
- GPT 3.5 can’t use images directly, making it unable to directly assess cozmo’s environment
- Utilize Visual GPT (GPT 3.5) that can leverage external models to accomplish task for itself and assess cozmo’s environment
- Due to intermediary task of GPT, hard to reason about ideal model combinations to accomplish goals

Visual GPT

- Developed by Microsoft (repo can be found here)
- Connects GPT 3.5 with Visual Foundation Models
    - Does not directly send and receive images since this is outside of capabilities of GPT 3.5
    - Sends images to models via path to file and receives model outputs as text of path of image file
    - Uses external image models at its discretion that best fit task at hand
- Uses an LLM as an interface that provides general knowledge with Visual Models as "domain experts" to provide knowledge depth
- Uses pre-engineered preamble, format instructions, and suffix to regulate GPT's use of models and behavior
- Every model is presented as a tool with specific description and example use cases
    - GPT leverages these description to determine which tool would best fit its task
    - Make interchangeable, flexible, and scalable framework to create/utilize additional tools for specific purposes

Integration into Cozmo

- Remove extraneous classes and isolate models that we need
- Integrate classes and models with cozmo as nodes
- Make a centralized image holder class to prevent using linux file system for image storage
- Keep track of model calls to perform well with multiple classes
- Set up nodes in parallel to cozmo's actions that control calls surrounding GPT



Visual Model Selection

Attempted many combinations of agent types and models

More models means higher latency, more expensive computation, and more memory needed (the full visual gpt repo exceeds memory quota by significant amount)

Less models means less sophisticated approach and can lead to unreliable performance

Image Captioning is required default of visual gpt, so it is present in every approach (possibly in the background)

Solution 1

Details

- Agent Type: Conversational
- Models used: Image captioning, Image depth
- Default depth model was used (Intel/dpt-large from HuggingFace), which we stuck with for our initial attempts

Prompting: “What objects are in the robot.png and how far are they”

Motivation

This was an exploratory phase to see the capabilities of GPT high level control. It was a test to see whether

1) Depth would be automatically leveraged at the discretion of cozmo
2) Whether results are viable to achieve the goal

Problems

1) Doesn’t give accurate depth
2) Can’t cross compare multiple image results. For instance, can compare the depth map difference between two distinct frames
3) Does not reliably classify whether or not objects are “obstacles”.

Learned - GPT can leverage other models if prompted well, even if the intent expressed is implicit. This leads to a rudimentary form of ***task planning***, which will be examined later

Solution 2

Details

- Agent Type: Conversational
- Models used: Image captioning, Image depth

Prompting: “Imagine you are a robot, can you move forward by 120 millimeters?”

Motivation

This was an exploratory phase to see the capabilities of GPT high level control. It was a test to see whether

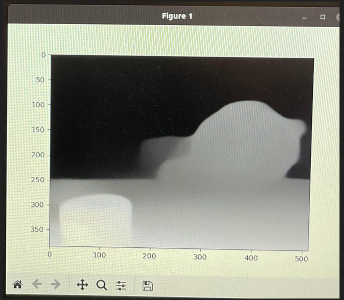
1) Give GPT context for wanting depth
2) Derive GPT results that are actionable outcomes, ‘yes’ or ‘no

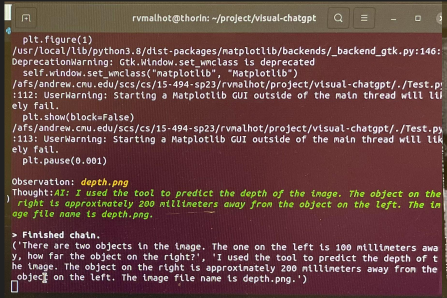
Problems

1) Still doesn’t use accurate depth
2) Unreliable errors where backend returns - ‘I can’t be a robot I’m an LLM’

Learned - Prompting needs to be clever, specific, and reliable to achieve results

Solution 3

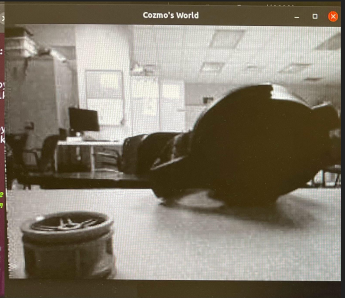




```
  plt.figure(1)
/usr/local/lib/python3.8/dist-packages/matplotlib/backends/_backend_gtk.py:146:
DeprecationWarning: Gtk.Window.set_wmclass is deprecated
  self.window.set_wmclass("matplotlib", "Matplotlib")
/afs/andrew.cmu.edu/scs/cs/15-494-sp23/rvmalhot/project/visual-chatgpt/./Test.py
:112: UserWarning: Starting a Matplotlib GUI outside of the main thread will lik
ely fail.
  plt.show(block=False)
/afs/andrew.cmu.edu/scs/cs/15-494-sp23/rvmalhot/project/visual-chatgpt/./Test.py
:113: UserWarning: Starting a Matplotlib GUI outside of the main thread will lik
ely fail.
  plt.pause(0.001)

Observation: depth.png
Thought:AI: I used the tool to predict the depth of the image. The object on the
 right is approximately 200 millimeters away from the object on the left. The im
age file name is depth.png.

> Finished chain.
('There are two objects in the image. The one on the left is 100 millimeters awa
y, how far the object on the right?', 'I used the tool to predict the depth of t
he image. The object on the right is approximately 200 millimeters away from the
 object on the left. The image file name is depth.png.')
```



Details

- Agent Type: Conversational
- Models used: Image captioning, Image depth

Prompting: “The object in the left corner is 40 millimeters away, how far is the the object on the right”

Motivation

We know that cozmo can’t use two images during two distinct time stamps to derive new results and conclusions as stated in solution 2. This was to see whether it would be possible derive conclusion by performing multiple inferences within the same image/timestamp.

1) Give GPT reference to make multi-step inferences
2) Check whether reference object improves overall depth perception

Problems

1) Uses accurate depth only sometimes
2) Unclear as to what operations are being performed in the background to derive the depth results, besides generating a depth map

Learned - GPT has capabilities to make multi-step inferences within the different models and the same model provided its within the same timestamp. Previous images can be loaded in still and new prompts regarding it can be made, but they need to be specific. For example, can’t load a depth map and ask for queries on it unless it was generated within the same timestamp

Solution 4 (Lift becomes reference)

Details

- Agent Type: Conversational
- Models used: Image captioning, Image depth
- Lift is set to partially cover top of cozmo’s camera view
-

Prompting: “The object on the top center is 30 millimeters away, how far is the the object in the center”

Motivation

Solution 3 was interesting in the fact that we could get better real depth estimation using a reference object. Integrating it with cozmo’s lift would be ideal and moving it to the top would also ensure that cozmo can detect object on the ground. We could also use odometry to derive the distance of the lift.

1) In built reference object for convenience
2) Boost depth perception accuracy using lift as reference object

Problems

1) Not as accurate as solution 3, GPT has a harder time with top/bottom vs left/right
2) Unclear as to what operations are being performed in the background to derive the depth results, besides generating a depth map

Learned - GPT is flexible in its prompting and generally returns sensible responses. However, it returns it with a high level of confidence even though the value could be trivially wrong in its entirety.

Solution 5 (Move Lift to Bottom)

Details

- Agent Type: Conversational
- Models used: Image captioning, Image depth
- Lift is set to partially cover bottom of cozmo’s camera view

Prompting: “The object on the bottom center is 30 millimeters away, how far is the the object in the center”

Motivation

Solution 4 performed worse than solution 3, but it was more practical since the lift only moves vertically and is attached to the robot. However, solution 4 was unreliable since the shadows cast by the lift (and the fact that the bottom of the lift is dark) made depth map generation inconsistent and inaccurate. The natural continuation is to adjust the position of the lift to the other side.

Problems

1) Still not as accurate as solution 3
2) Meaning behind values produced by depth map are still ambiguous since it is still being generated in the background
3) Lift is not perfect reference since it is very close to camera and would sometimes blend in with the ground

Learned: We found a more serviceable reference for depth map generation that could potentially be incorporated in final solution.

Solution 6

Details

- Agent Type: Conversational
- Models used: Image captioning, Image depth
- Lift is set to partially cover bottom of cozmo’s camera view

Prompting: “The closest object is 30 millimeters away, how far is the the object in the center”

Motivation

1) Slight variation in prompting might produce better/more reliable results
2) Give GPT context on the closest object, so that it implicitly asks it to scale it with respect to the rest of the depth map
3) Boost depth perception accuracy using lift as reference object

Problems

1) Not as accurate as solution 4, GPT has a harder time with top/bottom vs left/right
2) Unclear as to what operations are being performed in the background to derive the depth results, besides generating a depth map

Learned - GPT replies with sensible sentences but insensible values with high degrees of confidence. Example - “The closest object is 30 millimeters away and the object in the center is 10 millimeters away”. This can’t be possible as we previously stated the closest object is 30 millimeters away.

Solution 7

Details

- Agent Type: Conversational
- Models used: Image captioning, Image depth, Visual question answering
- Experimented with many prompts to Visual question answering to see if the additional model can be leveraged to improve performance

Prompting: “There are two objects in the image. The first one is on the bottom center and is 30 millimeters away. How far is the object in the center in millimeters?”
“There are two objects in the image and the object on the bottom is 30 millimeters away. The depth map generated from the image contains pixels whose intensity is each in the range 0-255. How far is the object in the center?

Motivation

1) Slight variation in prompting might produce better/more reliable results
2) Give GPT context on the closest object, so that it implicitly asks it to scale it with respect to the rest of the depth map
3) Boost depth perception accuracy using lift as reference object
4) Visual question answering might be more useful than image captioning and boost performance, even if it makes queries more cumbersome

Problems

1) Not as accurate as solution 4, GPT has a harder time with top/bottom vs left/right
2) Unclear as to what operations are being performed in the background to derive the depth results, besides generating a depth map

Learned - GPT replies is sensible sentence but insensible values with high degrees of confidence. Example - “The closest object is 30 millimeters away and the object in the center is 10 millimeters away”. This can’t be possible as we previously stated the closest object is 30 millimeters away.

Solution 8

Details

- Agent Type: Conversational
- Models used: Image captioning, Image depth, Visual question answering, Image Segmentation
- Attempted to include image segmentation model to help identify objects in the image.

Prompting: “Give me the segments for each object and get the depth for the green object”

Motivation

1) Image segmentation might help GPT identify objects in the picture better
2) We have already determined using multiple inferences from different models in one time step is possible

Problems

1) Additional model made queries to GPT very slow
2) Performance did not improve by enough to justify the more expensive operations

Learned: It is possible to get GPT to cross reference the outputs of multiple models and make layered inferences about the image. However, this could only be done with very specific prompts, and would probably not be realistic given our timeframe.

Solution: Conversational vs Zero-Shot Agent Type

Details

- Models used: Image captions, Image depth, Visual question answering

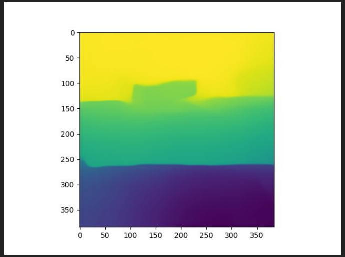
Prompting

1) Step 1: Multiple prompts of the same style were given to few shot learn: “There are two objects in the robot.png. The object on the bottom center is 30 millimeters away. The object in the center is {insertDistance} millimeters away”
2) Step 2: Legitimate prompt without the correct answer was provided to GPT. “There are two objects in the robot.png. The object on the bottom center is 30 millimeters away. How far away is the object in the center in millimeters”

Results

3) Conversational: Creates errors when given multiple/similar examples from user. Unable to learn corrections in expected outputs efficiently. Performs regularly and leverages models accurately regardless.
4) Zero-shot: GPT learns to zero-shot learn on its expected outputs. It first few responses during the training phase were “the difference in distance between the two objects is X millimeters”. This was eventually corrected to “The first object is 30 millimeters away and the second object is X millimeters away. However, it cannot enforce zero learning/task planning intent upon other independent models it leverages. This makes it impractical for image learning. The independent models leveraged don’t have feedback correction.

Solution: Depth map optimization



Details

- Agent Type: Conversational
- Models used: Image depth
- Removed all models other than depth to reduce complexity in model selection, replaced with own tools to see if we can extract real distances from the depth map
- Switched from using Intel/dpt-large to generate depth map, in particular tried Intel/dpt-hybrid-midas and MiDaS Swin 2 Tiny

Motivation

1) Depth map accurately provide qualitative information (such as whether one object is farther than the other) but we still cannot extract real values.
2) Other models were faster than Intel/dpt-large, and the 2 we tried were optimized for Monocular Depth Estimation
3) We verified that the real distance was proportional to the inverse of a linear transformation of the values returned in the depth maps of Intel/dpt-hybrid-midas and MiDaS Swin 2 Tiny due to model description and experimentation
4) Other models were not really helpful in identifying obstacles or their distances, and we still needed a way to extract real distances

Problems

1) Glare, specks, and other marks along the floor significantly hindered model's performance and introduced a lot of noise
2) Variance in the depth values along the lift, our reference object, was also messy and could lead to volatile and inaccurate distances when we tried to manually calculate the values

Learned: Trying to manually extract distances from depth map can be viable option, and integrating our own tools into Visual GPT's "toolbox" is useful since they can be streamlined for our needs and are not as cumbersome as the models provided by Visual GPT. Using other depth models more suited for our task and hardware is most likely necessary.

Additional Methods

1) (Implemented in project) GPT latency: too big of a delay between robot action and GPT responses. Solutions in this project included removing extraneous classes and rerouting communications internally without using the linux file system
2) GPT latency: poll gpt responses every 2 seconds so that GPT images queued for a response aren't outdated. We also decreased the robot's velocity to accommodate the high latency
3) (Implemented in project) GPT costs: Keep track of currently executing actions in the queue. Only run GPT on "forward" moving actions to check for collisions. Intuitively, we wouldn't check for collisions when turning in place as cozmo would not even see the object in the camera view.
4) (Implemented in project) Setup reference object for depth map using lift: Best results (least camera view intrusion + accurate depth map) came from SetLiftHeight(0.35), SetHeadAngle(-7) and were derived experimentally. Didn't set lift on top since shadows negatively impacted depth perception model.
5) (Implemented in project) Wrap colored tape around lift to increase contrast between the lift surface and the ground, which made the lift a more reliable reference object

Final Approach: Setup

Cozmo Details -

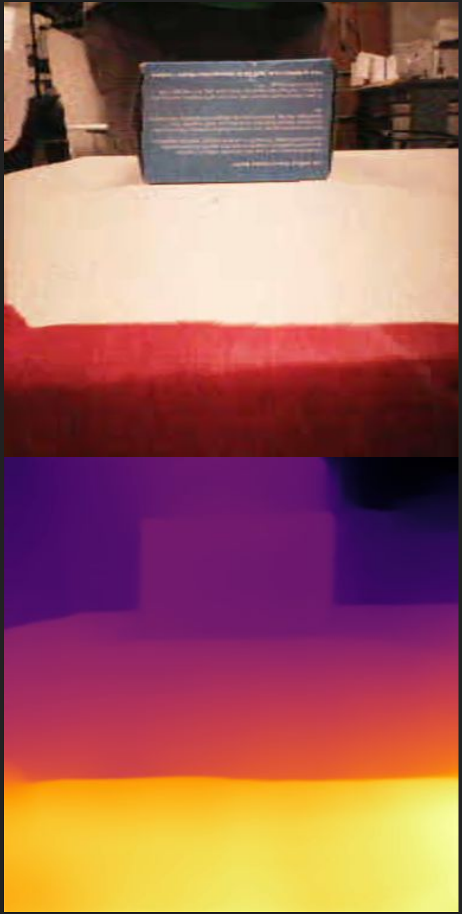
- Lift height: 0.35
- Head Angle: -7
- Image - RGB

GPT Details:

- Agent Type: Conversational
- Models used: Image captioning, Image depth, Real depth extractor

Final Approach: Depth model

- Cozmo RGB image is utilized
- Seam carving upscales the image without warping and stretching original image to reach better compatibility with depth model
  - Also accounts for fact that cozmo stores RGB images with half the number of pixels compared to grayscale images
- Processed image is fed to depth model MiDaS DPT Swin2 Tiny
- Output is stored in centralized class which GPT can use to intercommunicate within different models



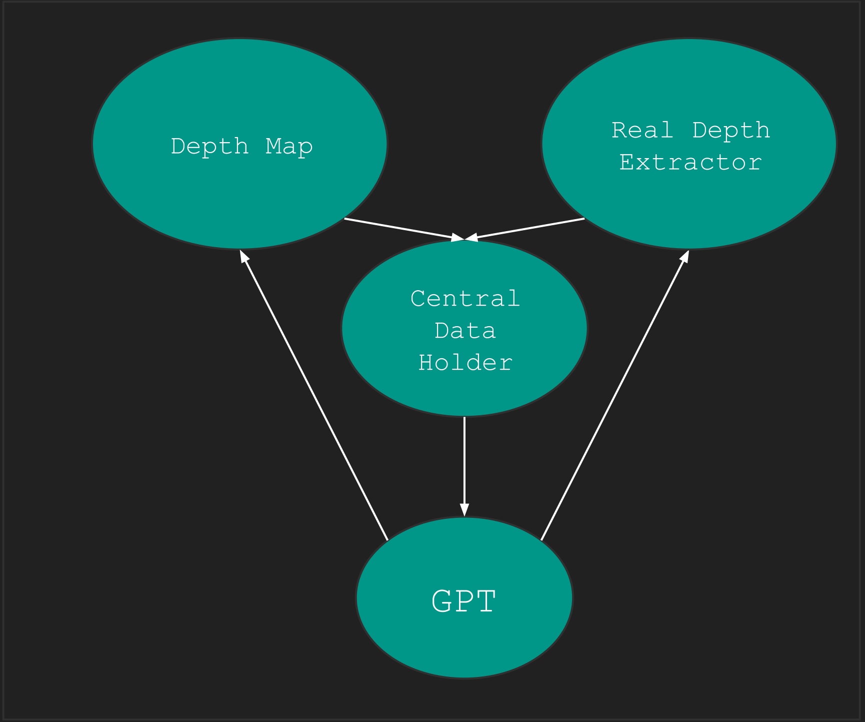
Final Approach: Real depth extractor

- Takes depth map
- Normalizes map
- Inverts map
- Uses average of bottom rows on the image (lift area) as reference to find scaling factor
- Scaling factor applied to entire depth map to find realistic depth

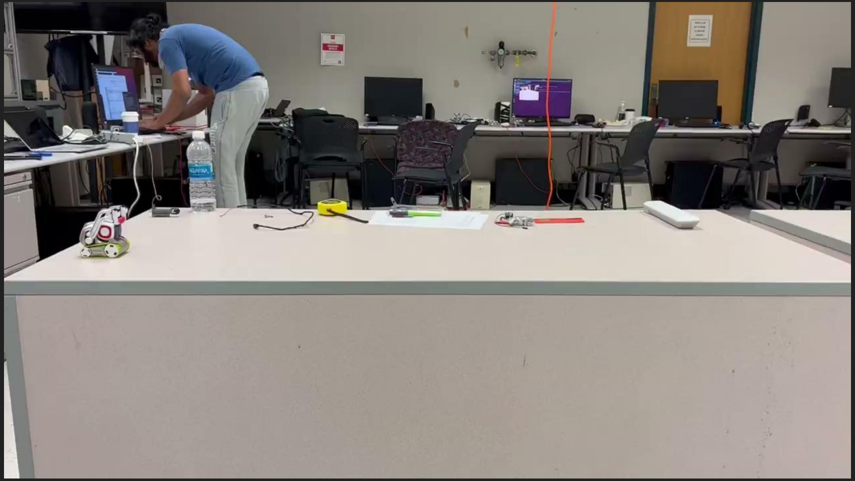
Final approach: Putting it together

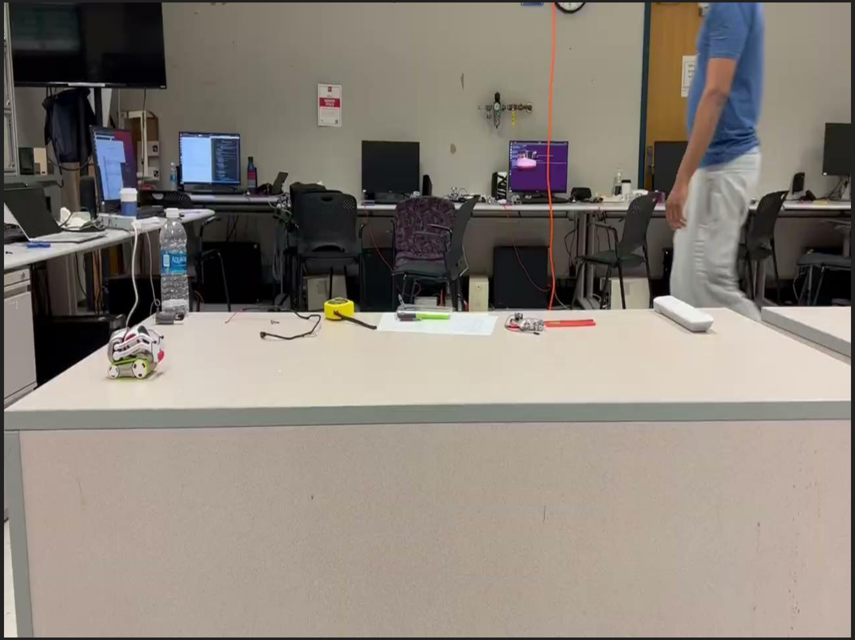
Prompt - “How far are the objects in robot.png and extract the real depth in millimeters ”

Explanation - We provide GPT with two tools. After seeing the prompt, GPT creates a Depth Map of the camera image. This is stored in a centralized class to organize the communication between GPT and other models. Then, GPT takes this depth map and asks the real depth extractor to give real depth of objects scaled in millimeters. The idea was to make GPT invoke different classes to show that it has multi-step inferences/reasoning to give a final answer to the prompt.

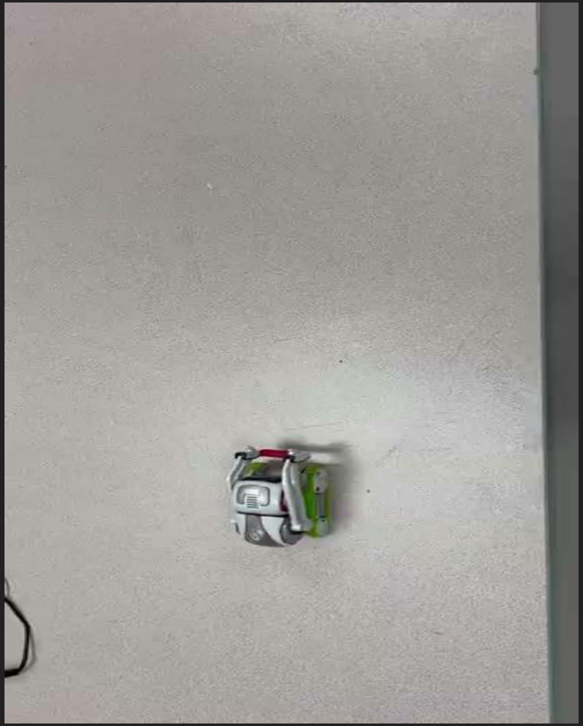


Video Demo (Control cases: no objects)





Video Demos (Abort sequence: object detected)



Future Extension

- Optimize every model GPT uses for robots and fast image processing
- Create less invasive abort sequence that could perhaps store existing state space
- Make rerouting plans and/or continuation plans
- Make more sophisticated inferences (ex: will the robot directly collide with object or manage to pass the object without course corrections?)
- Add more rich features such as doorways,etc that can be differentiated from objects in the way
- Extend more GPT control over other aspects of cozmo, such as task planning using multi-step reasoning