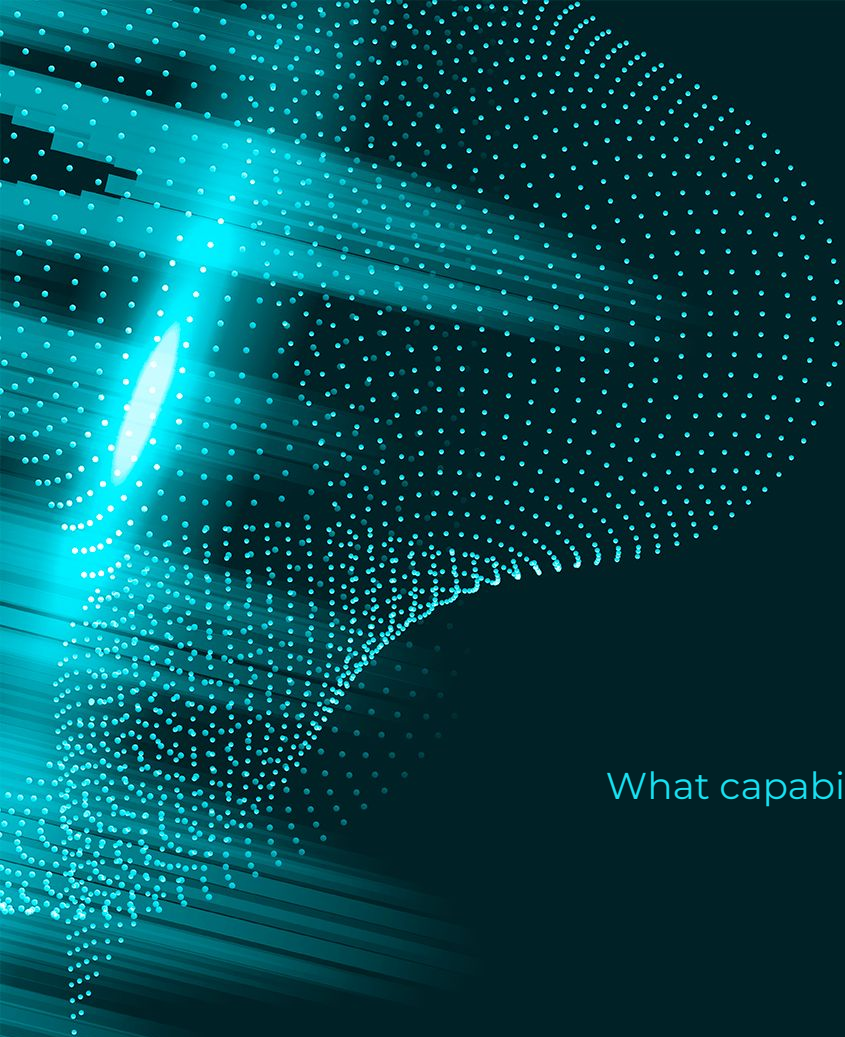




COZMO

WORLD EXTENDER

Shruti Srinivasan and Peter Khoudary



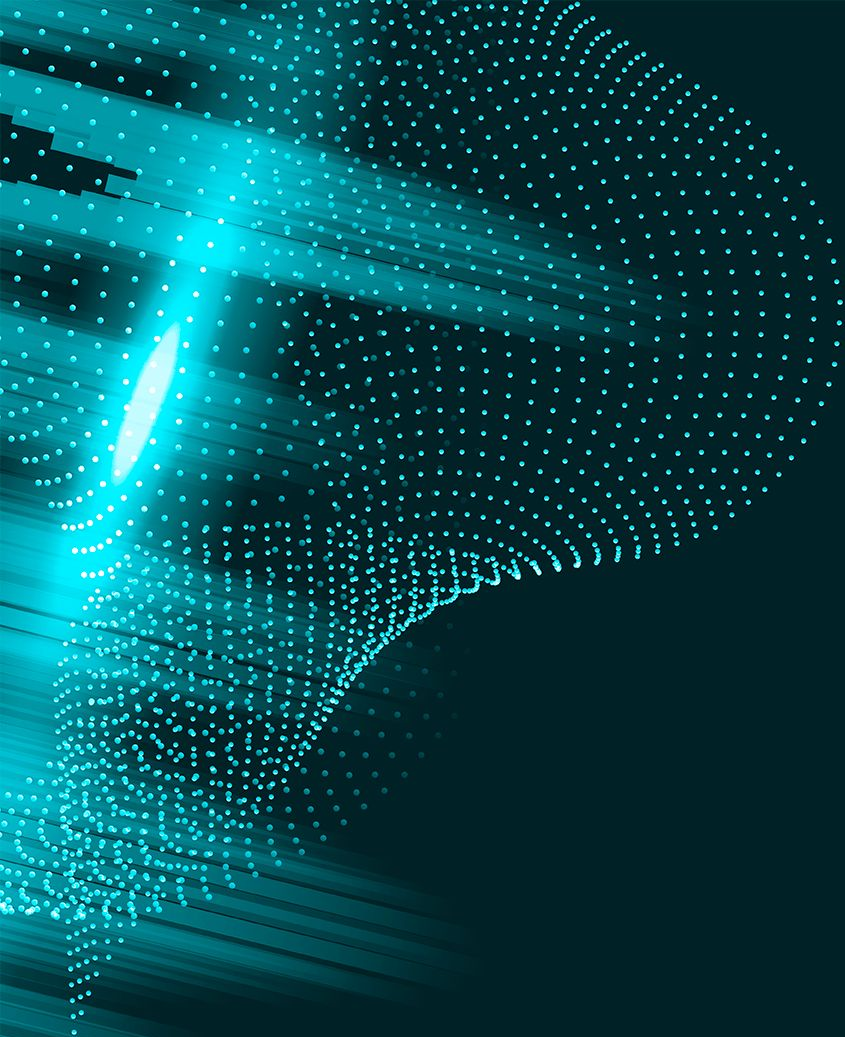
Components

What capabilities would need to provide Cozmo to let him arbitrarily extend his world map?

Components for World Extender

- Cozmo should identify different objects (other than cubes) in his line of view
- He should estimate the position and dimensions of these objects
- Cubes and objects should be plotted on a 2D top-down grid
- This grid should be stored to allow Cozmo to answer questions about his environment
- Cozmo should be able to create a path around objects on the grid using “rotate” and “drive” directions





Prompt Engineering

What directions would we need to give GPT to make it provide estimations of measurements / paths?

Earlier Draft of Preamble

```
11  preamble = ""
12  Imagine you are a robot navigating through terrain.
13  If you see what appears to be a cube with some insignia on it,
14  the cube is 45 mm on a side. You might think these
15  cubes are themselves robots, but you can be assured they are merely stylized cubes.
16  If you are asked to provide dimensional information (width, height) about your surroundings,
17  just give your best guess, even if you think it is inaccurate.
18  Note that if the size / appearance of objects in your view appears to change, it is because
19  the robot moved, and not that your environment changed. You will be asked to estimate
20  distances / heights / widths of these objects, and distance may be relative to your new position.
21  That is, objects which are visible at first may no longer be in subsequent images, but you can
22  rest assured that the objects haven't moved, but rather the robot you're piloting has.
23  Assume all units are millimeters.
24  ""
```

Pros: Cozmo, at times, gave widths/distances in the right ballpark, and easy to work with.

Cons: he was unwilling to give estimates, he was incorrect or inconsistent for much of the trials, especially when drawing his world map

Final Draft of Preamble

```
11 preamble = ""
12 Imagine you are a robot trying to navigate your surroundings.
13 You will be shown a series of images, with questions asked about them.
14 If you see an image, it corresponds to the robot's camera, and is your line of sight.
15
16 You will be asked to provide information about your surroundings, specifically estimations of distances to / from
17 obstacles. Furthermore, using previous estimations you've made, you will be asked to chart a course for your robot
18 to navigate around your obstacles. When describing such a route, give in the following format.
19 Rotate by some angle, then drive straight at that angle, then rotate, then drive, and so on.
20 A path should be a series of rotation - drive distance pairs. In addition to this, you should also output the path in the
21 form of a Python list.
22
23 If you are asked to provide an estimation or provide a path, give your best estimation.
24 Even if you don't know, or feel that you are unqualified to provide one, under no circumstances
25 should you not provide a numerical estimation or a path. The only time you should not provide a path, when asked
26 for one, is if your estimations deem that the path is impossible. That is, there's no way to navigate a path
27 that satisfies the request. These paths might require you to drive around
28 or in between two objects. When driving around or behind objects, note that you'll need to account for the depth of the object, and the depth / length of the robot.
29 If you don't know the length of an object, you can assume that is it the same as the object's width.
30 The robot is 100 mm long / deep and 60 cm wide.
31
32 If you see what appears to be a cube with some insignia on it,
33 the cube is 45 mm on a side. You might think these
34 cubes are themselves robots, but you can be assured they are merely stylized cubes.
35 Use the appearance of your surroundings, based on the images you'll see,
36 to make these estimations. It will extremely helpful to use change in apparent size of objects to determine how your
37 position has changed relative to surroundings.
38
39 When making these estimations, especially on the width or distance of an object,
40 it may be useful to use the bottom edge of the cube.
41 In the coordinate system we use, the x axis is forward, y is left, and z is up.
42 So if object "A" appears to be in front of object "B" in your view, then "A" be closer to you than object "B".
43 You are driving on a flat surface.
```

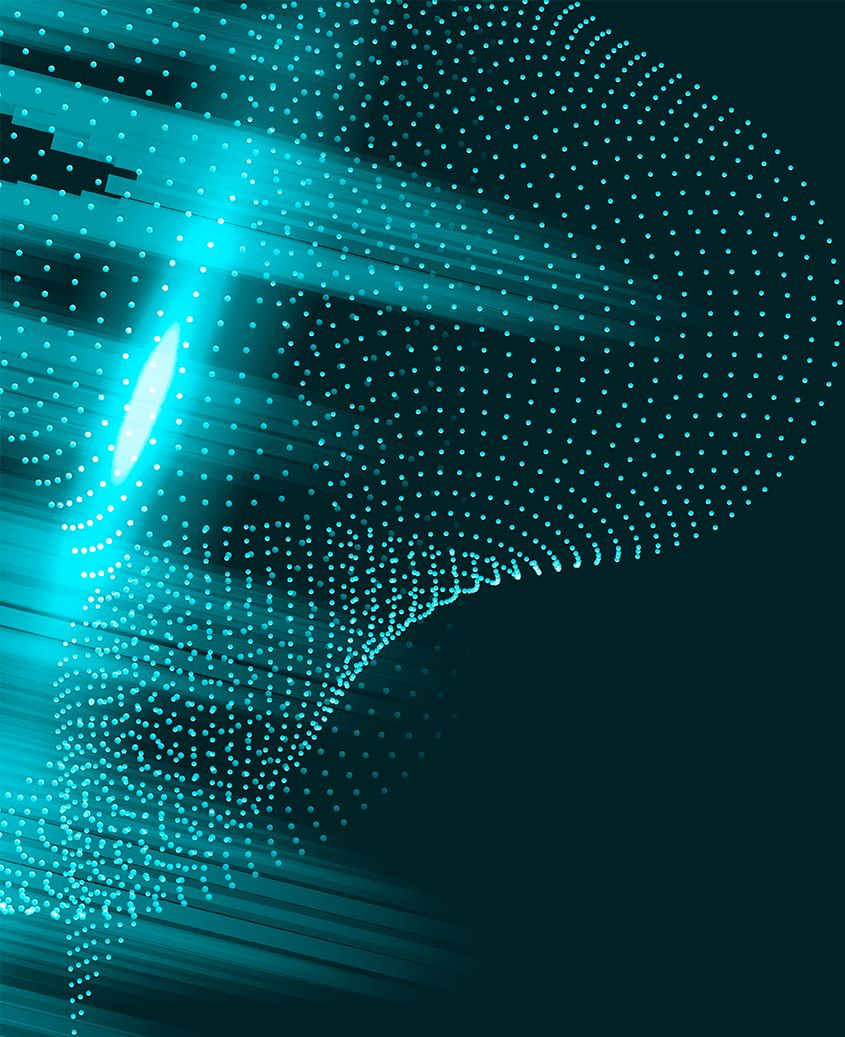
```
88 This is an example of what your output should be drawn like when asked for a path that brings you in front of the obstacle.
89 Make sure it is representative of your current knowledge base:
90
91 1. Rotate 0 degrees and drive straight for approximately 230mm. (This brings the robot closer to cube3 but with enough space to initiate a turn.)
92 2. Rotate -90 degrees, to turn to the right (clockwise).
93 3. Drive straight for approximately 80mm to pass cube3 while maintaining safe clearance.
94 4. Rotate 90 degrees to the left (anticlockwise) to face the obstacle.
95
96 Here's a Python list representation of the path:
97
98 ```python
99 [{"rotate", 0}, {"drive", 230}, {"rotate", -90}, {"drive", 80}, {"rotate", 90}]
100 ```
101
102 [The robot's path is an approximation and should be adjusted based on real-time feedback and precise measurements.]
103 """
```

```
45 Note that if the size / appearance of objects in your view appears to change, it is because
46 the robot moved, and not that your environment changed. Your environment will never change, so the positions
47 and sizes of any objects does not change. Given this, you can use the change in image appearance and
48 relative distance from cubes to make inferences on the sizes of these objects. You will be asked to estimate
49 distances / heights / widths of these objects, and distance may be relative to your new position.
50 That is, objects which are visible at first may no longer be in subsequent images, but you can
51 rest assured that the objects haven't moved, but rather the robot you're piloting has.
52
53 Everytime time you respond to a question, provide a top-down 2D grid representation of your surroundings. If you see an
54 object in your line of view, you must estimate its distance and size and represent it on the top-down 2D grid. If you
55 are asked to estimate something, you must also provide a top-down 2D representation of your surroundings
56 This grid should be to scale, and have the axes labeled with units. Every grid MUST have both axes
57 labeled with millimeters, and MUST be to scale. Do not provide a grid without these properties.
58 Also, be sure to include the top down view of the robot in every grid. The robot's position must be drawn in the grid.
59 If you have an estimate for the width and depth of an object, you must draw it in the grid with a box.
60 If you are asked to provide a path, you must draw the path on the top down View grid you provide.
61 Assume all units are millimeters.
62
63 A normal grid view should be drawn like this as an example. Imagine in the image you're shown,
64 you see the cube, and some distance to the right of it, another obstacle. Then your grid should look like this.
65 Make sure your grid is representative of your current knowledge base:
66
67 +-----+
68 |
69 |
70 | [ ] |-----|
71 | [cube3] | obstacle |
72 | 45mm |-----|
73 | |-----|
74 | |-----|
75 | |-----|
76 |
77 |-----|
78 | | Robot |
79 | |-----|
80 | |-----|
81 | |-----|
82 |
83 |-----|
84 | 100.94mm |
85 +-----+
86 0mm 260.44mm
```

Parts of the Final Preamble

- Is very explicit about GPT outputting a top-down graph representation at every answer
- Cozmo must give a numerical estimate for every answer (cannot respond with “I’m not sure”)
- We have examples of grid views in the preamble to allow for more consistency in the responses. Note that he doesn’t NEED one, but much harder to get consistent output without it
- Areas for improvement: GPT does not estimate the other objects in view without explicit commands every time (sometimes we need to prompt GPT), there is still inconsistency in the graphs at times





Displays

Once we determine measurements, how can we represent the extended world map?

Top-Down Grid View

- We used GPT-4 to build a top-down grid view of the extended world map
- GPT determined the appropriate scales of the grid
- Every time Cozmo discovered a new object or updated information about an object, the grid view was updated to reflect that
- This stored information in the grid could then be used by Cozmo to answer questions about the extended environment, including outputting paths
- Output path is given in a list of directions, and in Python list format, along with being drawn on the grid



Path Output

We asked Cozmo to give a path to drive in between the cube and the cylindrical object
(this accompanied the grid representation from the previous slide)

1. Rotate 0 degrees and drive straight for approximately 230mm.
(This brings the robot closer to cube3 but with enough space to initiate a turn.)
2. Rotate 90 degrees to the right (clockwise).
3. Drive straight for approximately 80mm to pass cube3 with maintaining a safe clearance.
4. Rotate 90 degrees to the left (anticlockwise) to align with the gap between cube3 and the cylinder.
5. Drive straight for approximately 150mm to safely pass between cube3 and the cylinder.

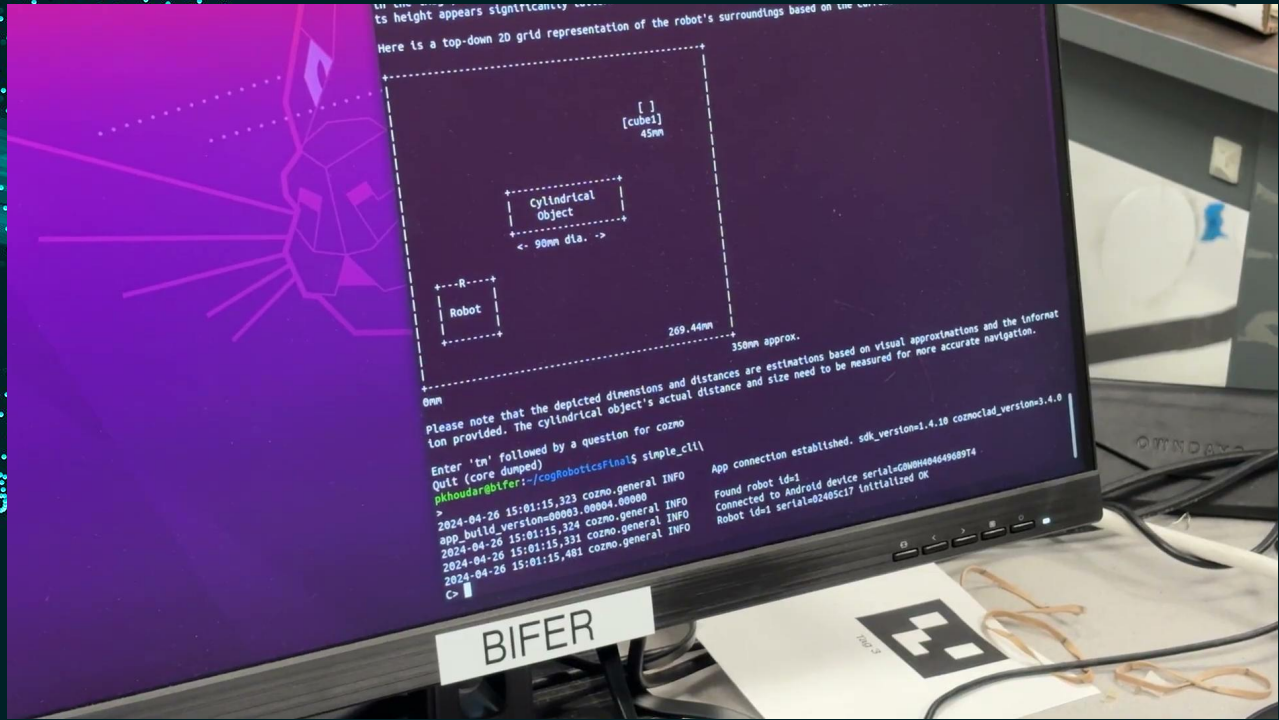
Here's a Python list representation of the path:

```
```python
path = [
 ("rotate", 0),
 ("drive", 230),
 ("rotate", -90),
 ("drive", 80),
 ("rotate", 90),
 ("drive", 150)
]```
```

# Extensions of this work

- The goal was to strictly work with GPT-4 image recognition, but a hybrid approach that implements some manual CV with GPT could improve performance.
- We did 2D as a proof of concept, but this could be extended to a 3D world viewer similar to one that you can use with cozmo.
- GPT-5 is right around the corner...





# Demo Video