LLM-Based Zero-Shot Object Navigation

Resources

Article: Dorbala, Can an Embodied AI Find Your "Cat-Shaped-Mug"? LLM-Based Zero-Shot Object Navigation

Introduction

Goal

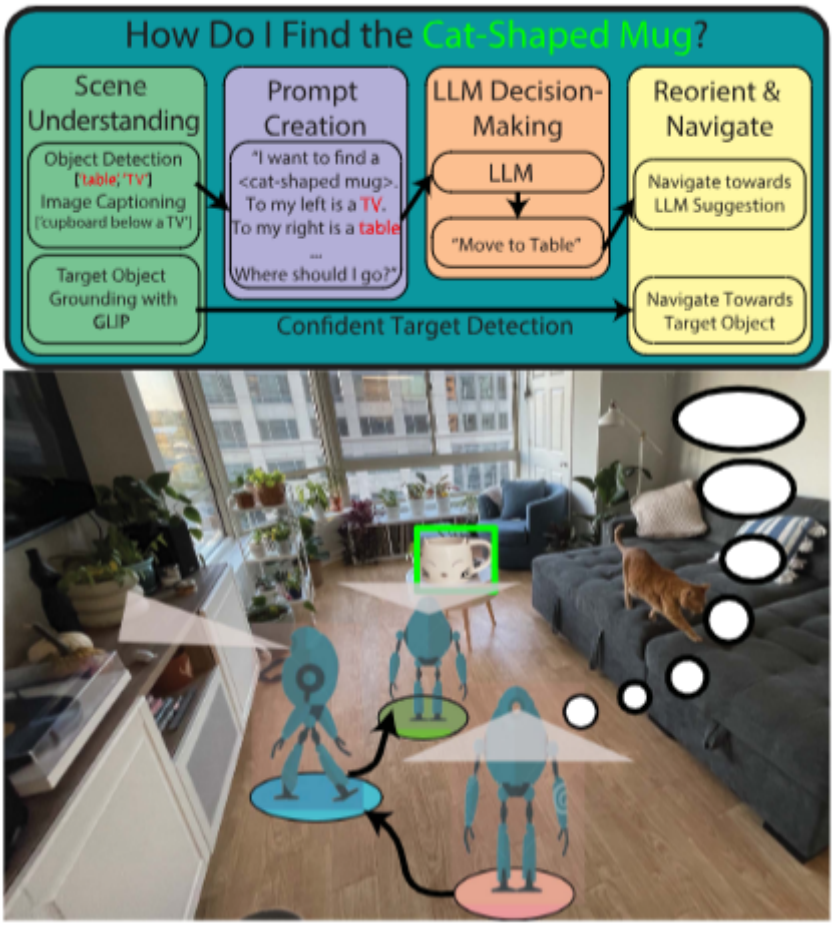
Find a **uniquely described target** in a **previously unseen environment**.

LLM-Based-Zero-Shot-Object-Navigation_Dorbala, page 1

Problem

- **AI is trained on images** it sees in its surroundings.
- **Real-world application** requires the robot to navigate to its goal, given the **description of what the goal looks like**.
- Often the robot needs to to navigate in **unknown environments** to find objects it has **not seen before** or are not common in daily-life, such as *cat-shaped mug*.
- AI needs to figure out what the goal object must look like, from the text description and navigate to find it.

Solution



The article suggests **Language-Guided Exploration (LGX)** to overcome the challenge of finding a *uniquely described target* in an *unknown environment*.

- **Large Language Models:** The commonsense and reasoning capbility of LLMs are used to make sequential navigation decisions.
- **Vision-Language grounding models:** Generalized target-object detection (simultaneously with the LLM navigation sequence generation). A Vision-Language model *bridges the gap* between what the robot sees in the surrounding and the description of what the robot wants to find.

Language-Based Zero-Shot Object Navigation

> ⓘ **LLM-Based Zero-Shot Object Navigation**
>
> - **LLM-Based:** The approach uses large language models to complete the task.
> - **Zero-Shot:** The agent is able to find objects it has seen *zero* times previously in environments it has navigated *zero* times previously. *Comes from the generalized idea of n-shot learning*.
> - **Object Navigation:** Given the description of the *object*, the agent needs to *navigate* to it.

In the article, this is the method proposed, used to approach the general class of problems known as **Language-Based Zero-Shot Object Navigation** or L-ZSON, where an agent needs to *navigate to a object (goal)*, given the *free-form natural language description* of that object in a *zero-shot* manner, having seen neither the *environment* nor the *object*, ever beforehand.

LLM-Based-Zero-Shot-Object-Navigation_Dorbala, page 1

Breaking it down

L-ZSON consists of two parts:

Sequential Decision Making	Target Object Grounding
Exploratory decisions on-the-go	Locate and *ground* (nail-down) the target object from *agent-perception* (what the agent sees)

LLM-Based-Zero-Shot-Object-Navigation_Dorbala, page 1

Existing Technology and Challenges

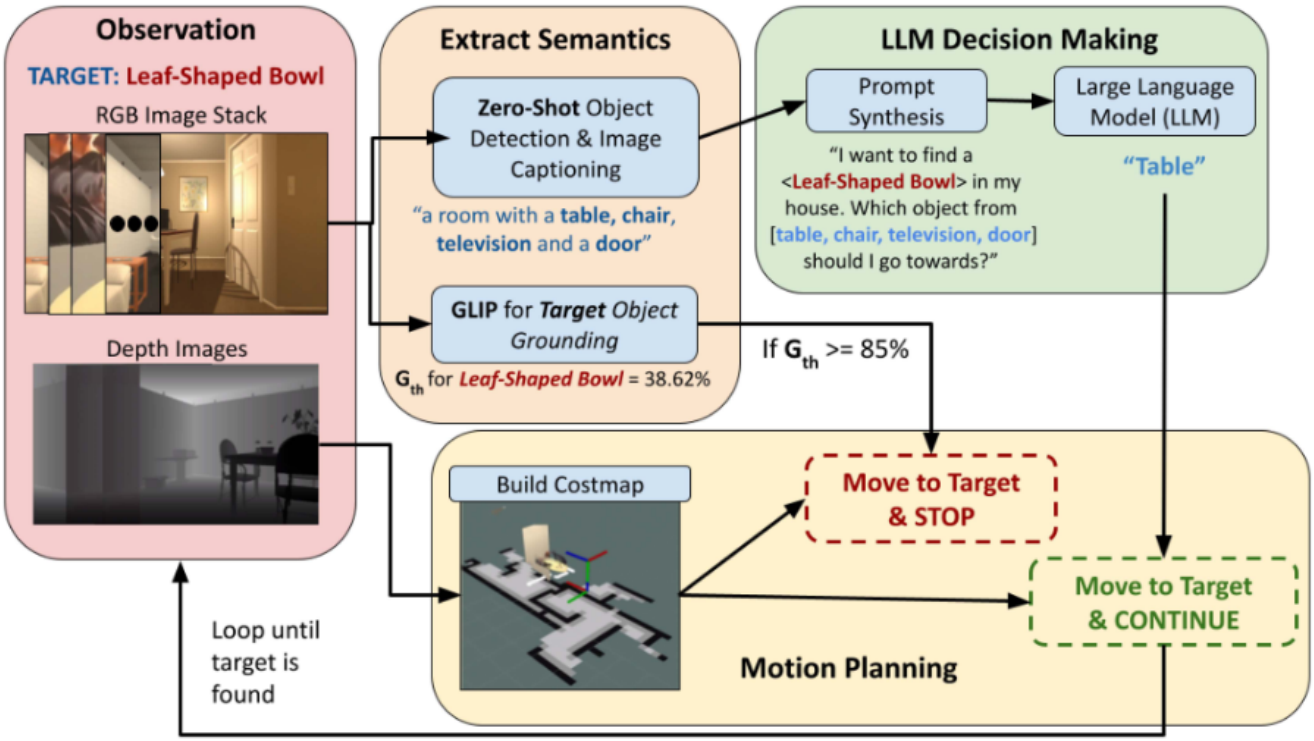
- State-of-the-art approaches are totally based on **supervised learning**.
- Not practical for performing consistently in dynamic real world environments, and detecting **arbitrarily-described objects**.
- Recent works:
  - Some recent works include attempts made to **generalize to locations** by removing environment-specific training.
  - Fewer recent works address the issue of **generalizing target grounding to novel-objects** (not present in the image datasets these models were trained on).
- None of these study **real-world test cases** that contain unconstrained language (humans can interact with the agent freely using whatever kind of language they want).
- These works utilize **large-scale pre-trained models** like CLIP and GLIP to perform zero-shot open-vocabulary target grounding in the wild
- *Downstream transfer* of these "foundation models" has improved the open-vocabulary *target grounding* in L-ZSON tasks.
- However, transferring foundation models for use in the sequential decision making component of L-ZSON is more difficult -> Some **experiential decision making** is required as the agent continuously interacts with the environment.
- **Challenges:**

- Exploiting the implicit knowledge contained in these large pre-trained foundation models to compose robot actions is a challenging task.
- Performance evaluation is difficult due to lack of adequate simulation environments. Currently available contain *common household items*, unable to provide evaulation over the *freeform natural language* descriptions humans use.

Contributions

1. Tackles **L-ZSON** by localizing (finding the location of) objects described by unconstrained (freeform) language by making use of large **Vision-Language Models** (VLMs), leveraging the **semantic connections between objects built into large langauge models.**
2. Utilizes *visual scene descriptions* to *formulate prompts* for LLMs, the output of which drives the navigation scheme. The study of various types of prompts provides insights into successfully using these prompts for robot navigation.
3. **27% improvement on SotA** *zero-shot* success rates (SR) and success weighted by path length (SPL) on *RobotTHOR*.
4. Successful transfer and evaluation of method onto real-world robots and study of carious complexities involved.

Methodology



Overview

Navigation

1. Extract **contextual cues** from scene (what the agent can view) in the form of object labels or scene captions.
2. Either of these cues are used to engineer a **prompt**, asking the LLM *"Where should the agent proceed to explore, next (in the environment)?"*
3. The LLM uses its **common-sense knowledge about object-relationships in the environment** to provide the agent with a direction to move toward.

Object-Grounding

1. A **Vision-Language Model** (*GLIP* is used here) is used to obtain a target object grounding score which gives us a **confidence score** of the *presence of the target described by the user in the scene*.
2. Once the confidence goes beyond a threshold G_{th}, the agent assumes that the target object is now in view (*technically*, in its *egocentric view*, which is the perspective or field of view of the robot itself).
3. The agent is successful in finding the *target object* if it is in the agent's view while performing *rotate-in-place*.

Scene Understanding

Environment Observation

- During each run (iteration), the agent first observes the environment by rotating 360°, taking images at a set resolution r.
- We thus two sets of obtain $\frac{360}{r}$ images
  - RGB Images, I_r -> Used to form **contextual cues**
  - Depth Images, I_d -> Used to construct a **2D costmap** of the environment

Contextual Cue Generation

Each RGB image in I_r is passed through one of two models:

- **Object Detection:** Model outputs scores (probabilities) for common household items (classes). Gives us a list of objects O which are found in scene I_r. *YOLO is used here*.
- **Image Captioning:** The model produces descriptive captions C of the scene I_r being observed by the robot. *BLIP is used here*.

We choose either O or C to pass as part of the prompt to the LLM.

Importance of Rotation-In-Place

- Allows the agent to fully observe surroundings, giving the **LLM enough information** to make a full informed navigation decision from the agent's current position in the environment.
- Without this step, the agent would proceed towards seen objects over unknown space, even if **none of the objects in the scene are related to the target object**.
- For example, if the target object o_g is "blue pillow" and the agent is facing a *kitchen*, it might go ahead and explore the "microwave", "mug" or "sink", because it does not know about the "bed" or "sofa" which might be directly behind it.

2D Costmap Construction and Navigation

While performing the rotate-in-place, the agent uses the depth images I_d from the scene to construct a costmap of the environment. *[RTABMAP] (https://introlab.github.io/rtabmap/#:~:text=RTAB%2DMap%20(Real%2DTime,location%20or%20a%20new%20location.) is used here*.

> [RTABMAP] (https://introlab.github.io/rtabmap/#:~:text=RTAB%2DMap%20(Real%2DTime,location%20or%20a%20new%20location.) uses **visual correspondence** along with **depth information** to compute the 2D costmap.

Once the 2D costmap is constructed:

- The **navigation decision** made by the LLM is considered:
  - *Object* if O was passed
  - *Direction* if C was passed
- According to the navigation decision, we **re-orient the agent**.
- Then we choose a **random point from the costmap**, along the agent's egocentric field of view (choose a point on the costmap which the agent can see).
- **Path planning from current position to the chosen point** gives us a path avoiding the obstacles in the way, using the costmap. The agent then continues it's exploration by navigating to the next point along the planned path.

Target Object Grounding and Exploration Loop

> GLIP is used for target object grounding.

- During each rotation step that the agent takes, the collected RGB image is passed through GLIP along with the target object description o_g as a prompt.
- When the grounding accuracy (score) of the current image and o_g goes above a threshold G_{th}, the agent assumes that o_g has been found and triggers a `STOP` signal. -> The episode is rendered **SUCCESSFUL**
- If o_g is not found, the agent continues exploration till n_r number of *rotate-in-place* operations have been performed.

Intelligent Exploration with Language Models

As discussed in Contextual Cue Generation,

1. **YOLO -> LLM:**
    - YOLO -> List of objects O
    - O used to synthesize a **prompt**
    - Prompt passed to LLM to make **navigation decision -> OBJECT to go towards**
    - $r =$ lower value here
2. **BLIP -> LLM:**
    - BLIP -> Image captions C
    - Captions passed in prompt to LLM to make **navigation decision -> DIRECTION to go in**
    - $r = 90^\circ$ here to allow the agent to explore in `left, right, front, back` directions

> If output from LLM **does not match** any of the expected outcomes, the agent chooses a random direction and proceeds with the point choosing step discussed in 2D Costmap Construction and Navigation.