A Robot for Cleaning and Sorting Garbages in the Home Environment

Jiajun Long, Ximan Zhang, Yikai Zheng, Jie Luo, Vong Sinrithy

I. Introduction

The objective of this project is to focus on the cleaning of a room and the cleaning and sorting of garbage by a robot in a household environment, and to evaluate the results obtained. The objectives of this project are described in detail below.

In a defined home environment, randomly placed a number of household items, such as a sofa, chairs, tables, and one or more moving cats or dogs.There are one or more trash cans in this household environment. The robot of this project needs to identify furniture, moving animals and garbage when cleaning the overall environment. On the basis of avoiding the movement path of pets, the robot should throw one or more garbage which are randomly appeared on the ground into the corresponding trash can and then continues the sweeping work.

II. Problem Statement

A. object identification

The following types of items need to be judged differently, and the corresponding trajectory planning process should be made.

1) Household items: Household items mainly include furniture and other items that may be found in people's house, such as sofas, chairs, tables, etc. After identifying these items, the robot should avoid these obstacles by planning its trajectory. Based on the good SLAM mapping in the iGibson, the collision between robot and furniture is almost unlikely to happen. However, for the safety reasons, this project adds a reward associated with the collision. Once the robot collides with the objects, its reward will become smaller or even negative.

2) Dynamic items: In addition to household items, there are also moving dogs or cats in the household environment. Robots also need to identify and avoid obstacles during movement, and at the same time, they need to predict their next trajectory. In the iGibson platform, There is a module called DynamicNavRandomTask which can carry out trajectory planning for moving objects. This project tries to modify it from the basis, so that it can perfectly conform to the actual situation of our project.

3) Garbage classification: In this project, there is another key decision that needs to be made to sort the garbage and throw it into the corresponding trash can. Garbage will be divided into several categories, which is more convenient for our robot to identify and classify it. After grabbing the garbage, it needs to be quickly thrown into the corresponding trash can. It can be modified and improved by referring to point_goal_reward and reaching_goal_reward functions in its platform. As a result, the reward of the garbage grabbing robot increases substantially when it places the garbage in the corresponding bin, while the reward of the wrong bin decreases slightly. In terms of trajectory planning, iGibson also has a similar module called RoomRearrangementTask, which can be modified to meet our expectations. After the trash is thrown into the trash can, the robot continues its sweeping or mopping work nearby.

B. navigation

Navigation in indoor environments presents additional challenges due to space constraints that limit maneuverability around the pedestrians and create layout dependent patterns of pedestrian interaction. These spaces, such as corridors, doors and intersections pose hard problems for any existing navigation solution. For these types of spaces, there is a need for new interactive navigation models for tight spaces with constraints imposed by obstacles, walls, doors and other common structural elements, based only on on-board sensor information.

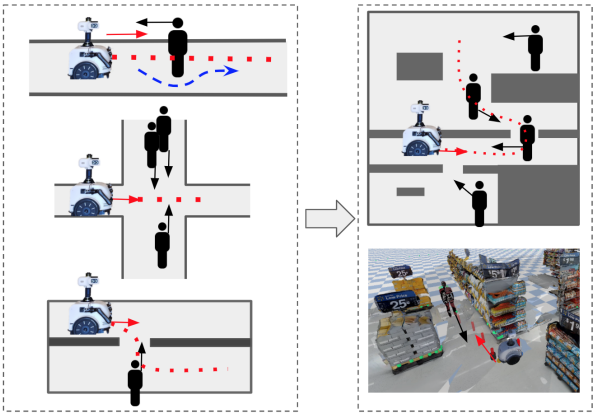

Fig. 1. Illustration of the problem of robot navigation around pedestrians in constrained indoor environments. The robot follows a global path (red dotted line) while maneuvering to circulate seamlessly around pedestrians (blue dashed line example). The robot's policy is trained on a small set of simple walls-world environments representative of common indoor layouts and the emerging interactions, such as in corridors and intersections (left). We show generalization to two types of unseen environments (right): more complex compositional walls-worlds (top right) and 3D reconstructions of real-world sites (bottom right).

In order to measure interactive navigation performance by measuring interactive navigation scores[1], this project refers to the evaluation criteria mentioned earlier and includes two aspects:

Path Efficiency: how efficient the path taken by the agent is to achieve its goal. The most efficient path is the shortest path assuming no interactable obstacles are in the way. A path is considered to have zero efficiency if the agent does not converge to the goal at all.

Effort Efficiency: how efficient the agent spends its effort to achieve its goal. The most efficient way is to achieve the goal without disturbing the environment or interacting with the objects. The total effort of the agent is positively correlated with the amount of energy spent moving its own body and/or pushing/manipulating objects out of its way. This aspect has been previously proposed to generate energy efficient sample-based motion planners

III. Literature Review

A. object identification

Object recognition technology is the foundation of the field of artificial intelligence. This project understands YOLO and its subsequent advanced versions through literature[2], introduces the development process of YOLO algorithm, and summarizes the methods of object recognition and feature selection.

In another paper[3], a hierarchical programming algorithm is proposed that can efficiently calculate an optimal plan for finding target objects in a large environment. In the house environment of this project, the increase in the number of moving operations and the number of objects leads to a significant increase in spatial complexity. In this paper, a good hierarchical programming solution is provided, effectively reducing a large problem to a small one by breaking it down into a set of low level in-container programming problems and a high level critical location planning problem utilizing low level programming. This project absorbs its experience and tries to apply it to the project.

B. mapping and navigation

The goal is to achieve interactive navigation,Since robots are more often deployed in homes and unstructured environments, such as homes and offices, it is not only inevitable but also necessary to consider the physical interaction part of navigation strategies. For example, when a robot navigates through a cluttered home, it may need to push objects aside or open a door to reach its destination. This article proposes a scoring method for interactive navigation[1],the Interactive Navigation Score, a novel metric to study the interplay between navigation and physical interaction of Interactive Navigation solutions.

Simulate interactive navigation in a narrow environment by referring to situations with pedestrian constraints[4], using reinforcement learning (RL) based methods to learn strategies that dynamically adapt to moving pedestrians when navigating between desired locations in a constrained environment. In this article, the learning method and simulation environment for robot oriented pedestrian navigation problems combine the ideas of reinforcement learning, planning, and multi-agent simulation to enable robots to navigate cooperatively in constrained environments that represent real indoor spaces. In this project, certain dynamic obstacle avoidance can be achieved through this method.At the same time, reference the interactive navigation scoring criteria to rate our project

IV. Model and Environment

The iGibson assets include the files required to build simulation scenes and other utilities, such as the URDF models for robots, models of a few interactive objects and articulated objects for testing, duplicates of the RBO and YCB datasets of objects to include them in the scenes, a replica of the Gibson V1 neural network filler, and a few mesh files for installation tests.

A. Model

In iGibson, there are 7 fully-supported robots (Table I). These robots inherit from 4 classes such as:
- LocomotionRobot: any robot with navigation functionality
- TwoWheeledRobot: any locomotion robot with two parallel wheels and differential drive functionality
- ManipulationRobot: any robot with one or more arms
- ActiveCameraRobot: any robot with a head or another controllable camera

TABLE I
List of 7 Fully-supported Robots in iGibson

Agent Name	Robot Type(s)	DOF	Information	Controller(s)
Mujoco Ant	LocomotionRobot	8	OpenAI Link	Base: {Torque, Velocity, Position}
Husky Robot	LocomotionRobot	4	ROS, Manufacturer	Base: {Torque, Velocity, Position}
TurtleBot	TwoWheeledRobot	2	ROS, Manufacturer	Base: {Torque, Velocity, Position, Differential Drive}
Freight	TwoWheeledRobot	2	Fetch Robotics Link	Base: {Torque, Velocity, Position, Differential Drive}
Fetch	TwoWheeledRobot ManipulationRobot ActiveCameraRobot	10	Fetch Robotics Link	Base: {Torque, Velocity, Position, Differential Drive} Camera: {Torque, Velocity, Position} Arm: {Torque, Velocity, Position, Inverse Kinematics} Gripper: {Torque, Velocity, Position, Binary}
JackRabbot	TwoWheeledRobot ManipulationRobot	2 & 7	Stanford Project Link	Base: {Torque, Velocity, Position, Differential Drive} Arm: {Torque, Velocity, Position, Inverse Kinematics}
LocoBot	TwoWheeledRobot	2	ROS, Manufacturer	Base: {Torque, Velocity, Position, Differential Drive}

As Fetch robot(Fig. 1) derives from TwoWheeledRobot, ManipulationRobot, and ActiveCameraRobot, Fetch robot has all of the capabilities of a real cleaning robot, so it will serve as the foundation for our model. However, we will alter it afterwards to make it more lightweight and resemble the actual cleaning robot.

Fig. 2. Fetch Robot

B. Environment

iGibson provides 4 types of scenes including:
- EmptyScene and StadiumScene: they are simple scenes with flat grounds and no obstacles
- StaticIndoorScene: they contain static indoor 3D scenes

- InteractiveIndoorScene: they contain fully interactive indoor 3D scenes

In interactiveIndoorScene, there are 15 fully interactive scenes(Fig. 2) which are similar to the household environment. Therefore, we will choose one amongst them and add some objects to be our environment, but for now, we will use either EmptyScene or StaduimScene because it is usefull for debugging purposes.

Fig. 3. 15 fully interactive scenes in the iGibson Dataset

V. Technical Approach

A. object identification and make policy

For this project, we mainly used an interactive iGibson environment. It is a simulation environment (based on Bullet) that provides fast visual rendering and physical simulation. It contains datasets of hundreds of large 3D environments recreated from real houses and offices, as well as interactive objects. iGibson allows researchers to use RGB images and other visual sensors to train and evaluate robotic agents to solve indoor (interactive) navigation and mobile operation tasks. iGibson implements all the functions needed to evaluate an AI solution in a BEHAVIOR benchmark: sampling logical activity descriptions, checking logical states, connecting to a BEHAVIOR dataset of 3D objects, and evaluating, which is a very powerful simulation environment.

At the same time, a large number of data sets are integrated into the environment and a wide variety of objects can be imported into the emulator. In the daily object dataset they created, 1,217 object models were curated using 391 object categories to support a wide variety of scenarios. Moreover, all the models are obtained by convex decomposition using VHACD algorithm, which makes collision detection more effective. It is known that both deep learning and reinforcement learning belong to the category of machine learning; Deep learning is labeled, static, and used mostly for perception. Reinforcement learning is unlabeled, dynamic and used for decision making. In this project, both machine learning methods are involved, which are respectively applied to object recognition and policy making.

Starting with object recognition, iGibson can be used with any learning framework adapted to the OpenAI gym interface. So there's a huge variety of algorithms that can be used in object recognition.

In the example of iGibson, the most classic CNN method is used. Firstly, the image is identified, and torch.nn is used to preprocess the image, and the preprocessing result is converted into neural network, and then CNN is used to transform it into convolutional layer for image feature recognition. Finally, the final classification was conducted by Multi-Layer Perception (MLP), and the results were obtained.

This project intends to use YOLO (you only look once) to replace the object recognition process in iGibson.

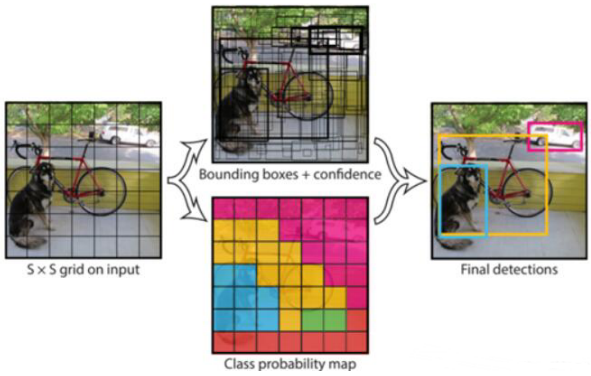

Fig. 4. YOLO Process

YOLO is an object recognition and location algorithm based on deep neural network. Its biggest feature is that it runs very fast and can be used in real-time systems. The purpose of YOLO is to find an object in a picture and give it a category and location. Object detection is based on supervised learning. The supervised information of each picture is the n objects it contains, and there are five information of each object, namely, the central position of the object (x,y), its height (h) and width (w), and finally its category.

There is also the classification problem. For each cell, different category probability values should be predicted. The convolutional network is used to extract features, and then the full connection layer is used to get the predicted values. The network structure refers to the GooLeNet model, including 24 convolutional layers and 2 fully connected layers, as shown in Figure 8. For the convolution layer, we mainly use 1x1 convolution for channle reduction, followed by 3x3 convolution. For the convolution layer and full connection layer, the Leaky ReLU activation function is used: $max(x, 0.1x)$. But the last layer uses linear activation functions. See Fig.4 for details.

Finally, the YOLO algorithm uses NMS first, then determines the categories of each box, and outputs the detection result only when its confidence value is not 0. That's the end of object recognition.

The above principle is the YOLOv1 algorithm, and the current YOLO algorithm has been heavily optimized and updated to v8. In this project, several advanced YOLO algorithms will be used to analyze the accuracy of object recognition, and the optimal method will be selected for subsequent operations.

After object recognition, the project also needs to make decisions about different objects and environments recognized. The project mainly uses PPO(Proximal Policy Optimization) and PPO2 On Policy reinforcement learning algorithms. Its implementation is simple, easy to understand, stable performance, can simultaneously deal with discrete and continuous motion space problems, conducive to large-scale training and other advantages, and the implementation of the two algorithms are compared, the accuracy and path effectiveness curve.

B. navigation

By controlling a simulated non-holonomic wheeled robot in multiple environments on an interactive Gibson simulator, which leverages the Pybullet physics engine for realistic physics simulation and supports virtual sensing (lidar) for our multi-agent setup. The environment includes simulated pedestrians driven by the Optimal Reciprocal Collision Avoidance (ORCA) model [4] based on a social forces model.

1) Reactive Navigation to Follow a Path: We propose to address the navigation problem in pedestrian environments with a combination of a sampling-base motion planner and a reactive low level sensorimotor policy learned with RL and implemented as a deep neural network.

We use the Soft Actor-Critic (SAC) algorithm to learn both a value function (critic) and a policy (actor) approximated by deep neural networks. SAC optimizes for a combined objective of the expected sum of rewards and the entropy. The policy network maps from observation space to action space, providing velocity references to a low level velocity controller. The reward function is composed of multiple terms that encourage the policy to reach the final goal, minimizing time, following the given global path but deviating from it to avoid the penalized collisions:

$$R = R_{goal} + R_{timestep} + R_{collision} + R_{potential} + R_{waypoint} \quad (2)$$

BehaviorTask: the agent should complete a long-horizon household activity defined in logic language

VI. Results

To evaluate the results, the first is to evaluate the object recognition, can evaluate the accuracy of household items recognition, get the average recognition accuracy. Similarly, the accuracy of garbage classification is evaluated.

For constrained navigation, we will use we will use different training methods to conduct training in different scenarios, and evaluate the training results of different methods and scenarios. The following metrics are used to evaluate the objective performance of the system. 1.Success rate (S): percentage of episodes where the robot reaches its goal without colliding or timing out; 2.Collision rate (C): percentage of episodes during which the robot collides with either a pet or static object such as a wall; 3.Collision with pets (PC): percentage of episodes involving collisions with pets; 4.Collision with obstacles (OC): percentage of episodes involving collisions with static obstacles such as walls; 5..Timeout rate (TO): percentage of episodes where the robot runs out of time before reaching its goal.

References

[1] Fei Xia, William B. Shen, Chengshu Li, Priya Kasimbeg, Micael Tchapmi, Alexander Toshev, Li Fei-Fei, Roberto Martín-Martín, and Silvio Savarese. Interactive Gibson Benchmark (iGibson 0.5): A Benchmark for Interactive Navigation in Cluttered Environments. *arXiv e-prints*, page arXiv:1910.14442, October 2019.

[2] Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, and Bo Ma. A review of yolo algorithm developments. *Procedia Computer Science*, 199:1066–1073, 2022. The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 & 2021): Developing Global Digital Economy after COVID-19.

[3] Yoonyoung Cho, Donghoon Shin, and Beomjoon Kim. ω^2: Optimal hierarchical planner for object search in large environments via mobile manipulation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7888–7895, Oct 2022.

[4] Claudia Pérez-D'Arpino, Can Liu, Patrick Goebel, Roberto Martín-Martín, and Silvio Savarese. Robot Navigation in Constrained Pedestrian Environments using Reinforcement Learning. *arXiv e-prints*, page arXiv:2010.08600, October 2020.