A Robot for Cleaning and Sorting Garbages in the Home Environment

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

***Abstract*—This paper mainly focuses on the task requirements of mobile robots identifying different kinds of garbage through object recognition in household environment, and can throw the garbage into the corresponding trash can. It respectively carries out in-depth exploration on the mapping navigation and object recognition of the robot, and combines the two in the later stage, and proves the feasibility of the task through three experiments: In object recognition experiment, the correct rate of object recognition in simulation environment can reach more than 90%. The similarity between the map generated in the navigation experiment with building map and the ground truth can reach more than 70%, and the effect is good in the integrated experiment. Finally, some solutions to the task limitation and prospects for the future are put forward.**

I. Introduction

A. Detailed Introduction

Today, with the booming service robot industry, the cleaning robot stands out as a very important part of it, and existing robots have already appeared in some people's house. The current cleaning robots mostly use SLAM to sense the environment, identify home environment through cameras to avoid collisions, and clean up the ground.

This project focuses on mobile robots similar to cleaning robots. It is hoped that on the basis of meeting part of the goals of cleaning robots, mobile robots can perceive the environment through SLAM in home environment and identify different kinds of garbage through object recognition algorithm. Secondly, it can be absorbed through the suction cup of the robot. Finally, the process of throwing the corresponding garbage into the corresponding trash can.

The objectives of this project are described in detail below:

In a defined home environment, randomly placed some household items, such as sofa, chair, table, etc. There is one or more trash cans in this household environment. The robot of this project needs to identify furniture and garbage during the identification and cleaning of the overall environment, throw one or more garbage randomly appearing on the ground into the corresponding trash can and then continue the cleaning work.

B. Related Work

1) SLAM and Navigation: According to the mathematical optimization framework used, laser SLAM can be divided into two categories: filter-based and graph-based laser SLAM. Filter-based methods include Hector SLAM, Fast-SLAM, Gmapping, etc. While graph-optimized methods include Karto SLAM, Large-SLAM, etc.

These methods all have their own advantages and disadvantages. In the "An evaluation of 2D SLAM technologies available in robot operating system"[1], the 2D laser SLAM effects under ROS were compared. The article mentioned that hectorSLAM is based on an optimized algorithm (solving the least squares problem), which does not require an odometer and can adapt to uneven air or ground conditions. However, the selection of initial values has a significant impact on the results, so it is required that the radar frame rate be high. The Gmapping algorithm adopts the RBPF method, which can build indoor environment maps in real-time. In small scenes, the computational complexity is low, and the map accuracy is high, requiring low scanning frequency for LiDAR. However, as the environment increases, the memory and computational complexity required for building maps will become huge, so Gmapping is not suitable for large-scale scene composition. The Sparse Pose Adjustment (SPA) adopted by Karto SLAM in the ROS environment is related to scanning matching and closed-loop detection. The more landmarks there are, the greater the memory requirement need, and the greater the advantages of other methods in mapping in the larger environment.

Based on the open-source mapping code in the ROS environment and combined with our proposed goal of building and navigating indoor small environments by a sweeper, Finally, we chose the Gmapping algorithm as our sweeping robot mapping algorithm.

2) Object Recognition: 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.

II. Dataset and Environment

A. Model

Fig. 1. Physical and simulation drawings

In terms of simulation model selection, this project has constructed the simulation model in the gazebo environment by referring to the TurtleBot3 Burger model, as shown in Fig. 1.

This project sets the robot to be $2kg$, with a radius of $0.15m$, and two wheels with a diameter of $0.064m$. The robot imitates the motion control system of TurtleBot3 Burger and uses the differential wheel to achieve omnidirectional motion. In terms of speed control, this project sets the maximum speed of the robot to be $0.5m/s$, the maximum acceleration of the two wheels to be $1m/s^2$, the maximum angular speed to be $1rad/s$, and the maximum angular acceleration to be $1rad/s^2$. We are equipped with IMU sensors to detect and measure acceleration, tilt, impact, vibration rotation and multi degree of freedom (DoF) motion make it easy to solve navigation, orientation, and motion vector control. Use LiDAR to detect the distribution of obstacles around the robot, achieving functions such as mapping and obstacle avoidance. The LiDAR is set to sample 500 particles and the detection radius is set to $0.1m$-$6m$. Equipped with a depth camera and RGB as object recognition tools, the camera's capture range is set to $10m$, and the captured photo pixels are 2560 $\times$ 1440. There is a vacuum scraper installed below the robot's site to absorb the recognized garbage, simulating the cleaning of the home environment by a cleaning robot.

B. Environment

In the simulation environment construction, this project used Gazebo to build a home environment, which can accurately and efficiently simulate the robot's working function in complex indoor and outdoor environments. Considering the robot's cruising time and the general size of the home environment, our environment size is $20m \times 10m$. In this environment, this project designed some obstacles to test the navigation and obstacle avoidance ability of the sweeping robot, as shown in Fig. 2, then divide the environment into three parts: living room, kitchen, and bedroom. In the living room, there are sofas, trash cans, tables, etc. There are dining tables and chairs as obstacles in the kitchen area, and fitness equipment, desks, etc. are set up as obstacles in the bedroom. At the same time, this project will add objects such as bowls, water bottles, water cups, beds, balls, tables and chairs, refrigerators, wardrobes, barbells, etc. to the simulation environment. These objects are used to verify the robot's object recognition ability.

Fig. 2. A simulation of house

C. Dataset

In this project, MSCOCO data set is mainly used, which is a large image data set developed and maintained by Microsoft, mainly used for target detection, target segmentation and image description, and mainly has the following characteristics:

- Object Segmentation
- Superpixel stuff segmentation
- 1.5 million object instances
- 80 object categories

In this project, all categories of coco data set were not used, but six items in the data set were mainly used for identification. The picture features of the data set were clear, which made it easier for training and testing, as shown in Fig. 3.

Fig. 3. The picture shows the six types of objects selected by the project for identification in the coco dataset, from left to right, and from top to bottom: giraffe, horse, cat, cup, bottle and bowl. The figure is one of the training pictures of these objects in the coco dataset.

Objects in this project are mainly divided into three categories. The first category is objects that need to be grabbed. In this project, bottles, bowls and cups are mainly grabbed. The second category is the auxiliary identification items of different trash cans. By pasting the images of giraffe, cat and horse on the trash can, different trash cans can be identified. This method is used to complete the auxiliary identification of garbage cans, avoiding the embarrassment of having no corresponding category of garbage cans in the data set. The third type of furniture independent of the above six kinds of garbage can also be identified, including tables, chairs, sofas, etc., but no other operations are made on them.

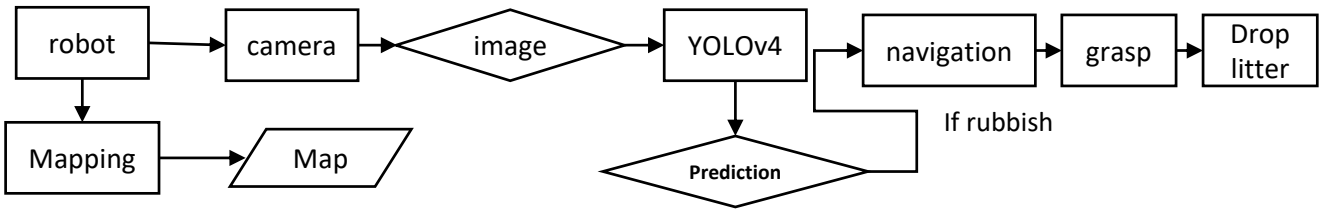


Fig. 4. This picture shows the overall work flow of the project. After the robot is in the home environment, it scans and builds a map for the overall environment, and saves the map. The environment is detected by the RGB camera of the robot, and the images captured by the camera are imported into the object recognition algorithm for recognition. The robot will autonomously navigate to the garbage place, grab the garbage through the suction cup, navigate to the garbage can according to the location of the garbage can, close the suction cup, and continue cleaning after the garbage is put down.

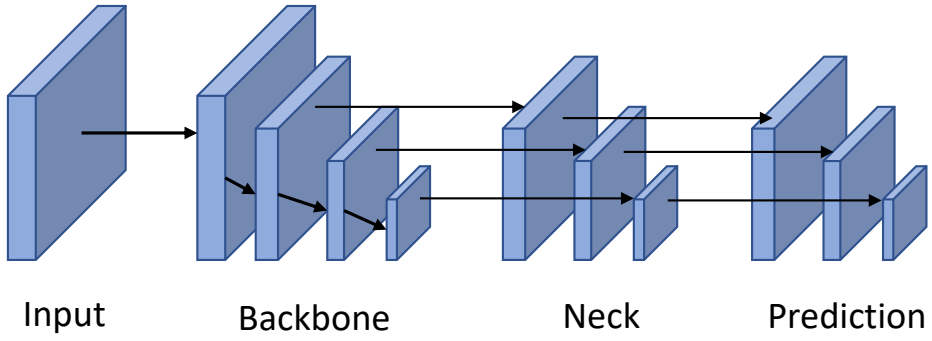


Fig. 5. The figure shows the whole process of YOLOv4 algorithm, which is mainly divided into four steps.

III. Methods

At the beginning, the overall work flow of this project is summarized, as shown in Fig. 4. The process of SLAM mapping, object recognition and navigation is combined. The respective steps are described below.

A. Object Recognition

1) Algorithm Framework: The following is an introduction to the object recognition algorithm framework used in this project.

YOLO (You Only Look Once) is an object detection method. By inputting the whole picture, detection information of multiple objects can be output, and the category and location of objects in the picture can be identified. In our project, we refer to and use YOLOv4 for object recognition[4]. YOLOv4 adds many practical techniques on the basis of YOLOv3, which greatly improves the speed and accuracy. The whole frame of YOLOv4 is shown in Fig. 5. This paper will interpret YOLOv4 from the Input, BackBone, Neck and Head output layers.

The first layer is the input, where 608×608*3 images are input for image preprocessing. Three algorithms are used to enhance this process. Firstly, Mosaic data enhancement algorithm is used to improve the training speed of the model and the accuracy of the network.The second is CmBN, which uses the ”information at the time of the current iteration” for normalization, and implements the operation of enlarging the batch size without additional compensation.The third is SAT (Self Adversarial Training) self adversarial training, which is a new data enhancement method. Using adversarial generation can improve the weak link in the learning decision boundary and improve the robustness of the model.[5]

The second layer is the BackBone, that is, the backbone network, uses the CSPDarknet53 network to extract features and simultaneously uses the Mish activation function, Dropblock regularization, and CSP cross-stage partial connection method. CSPDarknet53 is based on Darknet53 and using CSPNet[6] for reference, which make the model not only guarantees the reasoning speed and accuracy, but also reduces the size of the model, with strong accuracy in the field of target detection. Second, Mish function and ReLU, Swish and other activation function is very similar, but the different data sets can be in a lot of depth in the network have better performance, better accuracy and generalization of reference.[7] Here's Mish's function:

$$y = x \times \tanh((\ln(1 + \exp_x)) \tag{1}$$

The third is Dropblock regularization algorithm[8], which is used to solve the problem of model overfitting, mainly through

the deactivation of the local region.

The third layer is Neck, in YOLOv4 mainly added SPP module and FPN + PAN structure. The SPP module adopts the maximum pooling mode of 1x1, 5x5, 9x9 and 13x13 to carry out multi-scale feature fusion. Then, by using the method of FPN (Feature Pyramid Networks), the high resolution of low-level features and high semantic information of high-level features can be used to achieve the prediction effect through the fusion of features of different layers. Finally, Path Aggregation Network (PAN) was used to fuse the feature information of different size map to obtain accurate information.

The Head of YOLOv4 is the same as that of YOLOv3. The scale of prior frame is extracted by clustering and the position of prediction frame is constrained. Compared with YOLOv3, the loss function $CIOU_{Loss}$ during training and $DIOU_{nms}$ screened by prediction frame are mainly improved.

2) Executive Objective: This paper aims at the cleaning robot in the home environment, so the object recognition mainly includes the objects that may appear in the home environment, such as sofas, tables, chairs, etc., but also includes the garbage that we need to identify, such as cups, bottles and bowls. The primary goal is to ensure that the algorithm, after training through images in the real world, can accurately recognize objects in the simulation environment with high accuracy. Secondly, it is necessary to judge the position and direction of the object in the image representation scene.

B. SLAM and Navigation

1) SLAM: The general process of the Gmapping algorithm is to use the map and motion model from the previous time to predict the current pose, then calculate weights based on sensor observations, resample, update the particle map, and so on. It is based on particle filtering and can be roughly divided into the following four parts to complete. DrawFromMotion, ScanMatch, UpdateTreeWeights, Resample[9].

DrawFromMotion: The state of the particle at the current moment is first updated by the motion model, and Gaussian sampling noise is added to the initial value to perform a rough state estimation. Shown as Fig. 6

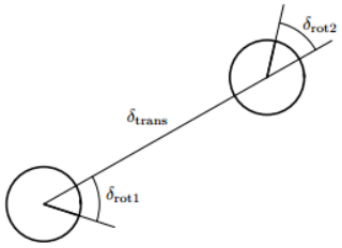


Fig. 6. Odometry model: The robot motion in the time interval [t-1,t] is approximated by a rotation rot1, followed by a translation trans and a second rotation rot2. The turns and translation are noisy.

Use the following algorithm to sample motion, shown as Fig .7

Algorithm sample_motion_model_odometry(u_t, x_{t-1})**:**

$\delta_{rot1} = \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$

$\delta_{trans} = \sqrt{(\bar{x} - \bar{x}')^2 + (\bar{y} - \bar{y}')^s}$

$\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$

$\hat{\delta}_{rot1} = \delta_{rot1} - \textbf{sample}(\alpha_1 \delta_{rot1} + \alpha_2 \delta_{trans})$

$\hat{\delta}_{trans} = \delta_{trans} - \textbf{sample}(\alpha_3\ \delta_{trans} + \alpha_4(\delta_{rot1} + \delta_{rot2}))$

$\hat{\delta}_{rot2} = \delta_{rot2} - \textbf{sample}(\alpha_1 \delta_{rot2} + \alpha_2 \delta_{trans})$

$x' = x + \hat{\delta}_{trans}\ \cos(\theta + \hat{\delta}_{rot1})$

$y' = y + \hat{\delta}_{trans}\ \sin(\theta + \hat{\delta}_{rot1})$

$\theta' = \theta + \hat{\delta}_{rot1} + \hat{\delta}_{rot2}$

return $x_t = (x', y', \theta')^T$

Fig. 7. First, obtain odometer readings to obtain motion control information,then add Error term to the variation of motion control, add this error to thepose at time t-1, and calculate the pose at time t

ScanMatch: Based on the predicted posture of the motion model, move the predicted posture towards six states: negative x, positive x, negative y, positive y, left rotation, and right rotation. Calculate the matching score for each state, and select the posture corresponding to the highest score as the optimal posture.

Algorithm likelihood_field_range_finder_model(z_t, x_t, m)**:**

$q = 1$

for all k *do*

if $z_t^k \neq z_{\max}$

$x_{z_t^k} = x + x_{k,\text{sens}} \cos\theta - y_{k,\text{sens}} \sin\theta + z_t^k \cos(\theta + \theta_{k,\text{sens}})$

$y_{z_t^k} = y + y_{k,\text{sens}} \cos\theta + x_{k,\text{sens}} \sin\theta + z_t^k \sin(\theta + \theta_{k,\text{sens}})$

$dist = \min_{x',y'} \left\{ \sqrt{(x_{z_t^k} - x')^2 + (y_{z_t^k} - y')^2} \,\middle|\, \langle x', y' \rangle \text{ occupied in } m \right\}$

$q = q \cdot \left(z_{\text{hit}} \cdot \textbf{prob}(dist, \sigma_{\text{hit}}) + \frac{z_{\text{random}}}{z_{\max}}\right)$

return q

After obtaining the optimal particle pose, the particle sampling range can be changed from the flat and wide area to the peak area L represented by the LiDAR observation model, and the new particle distribution can be closer to the real distribution. Shown as Fig .8

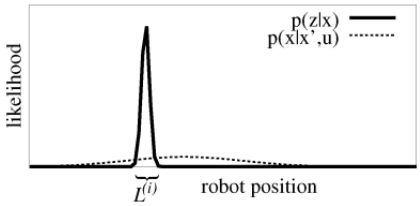


Fig. 8. The two components of the motion model. Within the interval L, the product of both functions is dominated by the observation likelihood in case an accurate sensor is used

By Scann and Match, the peak area represented by L was

found. After finding the peak area represented by L, it is necessary to determine the mean and variance of the Gaussian distribution represented by the peak area. By randomly sampling K points in L, calculate the mean and variance based on the odometer and observation model of these K points, as shown in the following equation.

$$\mu_t^{(i)} = \frac{1}{\eta^{(i)}} \cdot \sum_{j=1}^{K} x_j \cdot p(z_t \mid m_{t-1}^{(i)}, x_j) \cdot p(x_j \mid x_{t-1}^{(i)}, u_{t-1})$$

$$\Sigma_t^{(i)} = \frac{1}{\eta^{(i)}} \cdot \sum_{j=1}^{K} p(z_t \mid m_{t-1}^{(i)}, x_j) \cdot p(x_j \mid x_{t-1}^{(i)}, u_{t-1}) \cdot (x_j - \mu_t^{(i)})(x_j - \mu_t^{(i)})^T$$

with the normalization factor

$$\eta^{(i)} = \sum_{j=1}^{K} p(z_t \mid m_{t-1}^{(i)}, x_j) \cdot p(x_j \mid x_{t-1}^{(i)}, u_{t-1}).$$

UpdateTreeWeights: For each particle, we need to calculate its weight for use in subsequent resampling steps.The weight describes the difference between the target distribution and the proposed distribution. The formula for calculating the weight is

$$\begin{aligned} w_t^{(i)} &= w_{t-1}^{(i)} \cdot p(z_t \mid m_{t-1}^{(i)}, x_{t-1}^{(i)}, u_{t-1}) \\ &= w_{t-1}^{(i)} \cdot \int p(z_t \mid m_{t-1}^{(i)}, x') \cdot p(x' \mid x_{t-1}^{(i)}, u_{t-1})\, dx \\ &\simeq w_{t-1}^{(i)} \cdot \sum_{j=1}^{K} p(z_t \mid m_{t-1}^{(i)}, x_j) \cdot p(x_j \mid x_{t-1}^{(i)}, u_{t-1}) \\ &= w_{t-1}^{(i)} \cdot \eta^{(i)}. \end{aligned}$$

ReSample: Before performing resampling, the weight of each particle is calculated. Sometimes, due to high environmental similarity or measurement noise, the weight of particles in the near correct state may be smaller, while the weight of particles in the wrong state may be larger.

In the Gmapping algorithm, the author uses the measure of weight deviation to determine resampling.

$$N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} \left(\tilde{w}^{(i)}\right)^2},$$

The larger the Neff, the smaller the difference in particle weight. When Neff drops below a certain threshold, it indicates a significant difference between the distribution of particles and the true distribution. At the particle level, it appears that some particles are very close to the true value, while many particles are far from the true value. At this point, resampling happens.

2) Navigation: For the navigation package configuration of ROS, navigation is a 2D navigation package set that outputs target position and safe speed for mobile robots by receiving odometer data, tf coordinate transformation tree, and sensor data. According to the official structural framework diagram provided by ROS, the navigation package is divided into three parts: AMCL and Move-Base and Map-sever.(see Fig. 9)

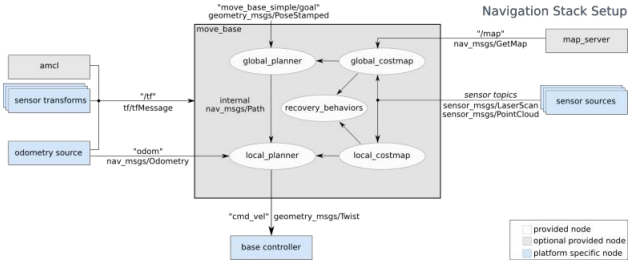


Fig. 9. Navigation framework diagram

Move-Base receives information such as odometer messages, robot posture and position, map data, and performs global and local planning within the node. Local planning is aimed at changing one's path according to changes in the environment during the navigation process, achieving automatic obstacle avoidance. AMCL achieves probability self-determination based on existing maps and LiDAR. It is used to confirm and output the robot's position in the current map. The principle of this positioning method is as follows: each sample stores position and direction data representing the robot's posture. Particles are randomly sampled, and when the robot moves, particles memorize the robot's actions based on their state, using recursive bayesian estimation for resampling. Map-Sever is used to load static global maps constructed by the Gmapping algorithm into a map server.

IV. Experiments

A. Object Recognition

In the first small experiment, the project focused on the mAP (Mean Average Precision) of the test set, i.e. the exact results. mAP is a commonly used evaluation index in target detection models. The first things to know are Precision and Recall. By precision and recall, AP can be known step by step. AP is for a certain category of all pictures. However, in the data set of this project, we are not only concerned with the identification of one category, so we need to use mAP to measure the quality of the model in all categories.

In terms of specific implementation, the original test code of YOLOv4 was modified first to obtain more parameters and results required by this project. The learned model was observed by observing mAP50, mAP75 and mAP95.

The experimental results are shown in the Fig.10 below.

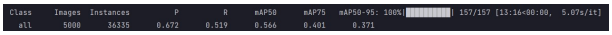


Fig. 10. The figure shows the final value obtained after multiple evaluations using YOLOv4 to validate the test set of the COCO dataset.

It can be found that in the experiment, mAP50 remained at 0.566 and mAP75 at 0.401, basically consistent with the results of YOLOv4 paper, indicating high accuracy of model training.

In the second small experiment, the probability that the object recognition is correct is judged by recognizing the object in the simulation environment. Since the training mainly uses images in the real world, this experiment is used to explore the accuracy rate of object recognition in the simulation environment, laying a foundation for the smooth progress of the later experiments. The main process is to put the three pieces of garbage selected in this project into the simulation environment, and directly let the camera on the robot acquire images for object recognition through YOLOv4, and judge the recognition accuracy. The following is part of the experimental results, as shown in the Fig. 11

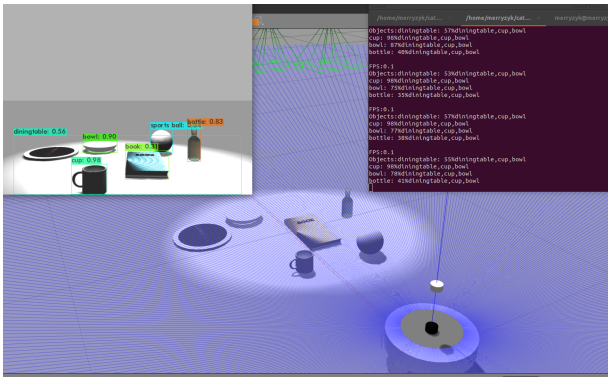


Fig. 11. The main purpose of this figure is to show the recognition accuracy of the object recognition model generated by training with real images on virtual items in the simulation environment. In addition to this figure, several sets of tests were performed, which are not listed here.

As can be seen from the pictures shown, it can be seen that the recognition of multiple items is still relatively accurate, among which the recognition accuracy of bowl, cup and bottle in this picture is high, reaching 80% or more, and the recognition rate of cup with the most obvious features even reaches 98%. Relatively speaking, book is difficult to identify accurately due to its shape. The recognition accuracy of the sports ball in the upper right corner is poor due to the light. Another object not recognized in the image is the frisbee in the upper left corner, which is difficult to recognize in gazebo images due to its shape and color. At the same time, there are some mistakes in recognition. There is no so-called dining table in the picture, but the white circle appears in the picture because the light source is too bright. But the light source is too dark and can't clearly identify other objects, which is one of the biggest problems encountered in the simulation environment.

In the third small experiment, this project hopes to compare with the object recognition algorithm popular at the same time in YOLOv4, and prospect the later YOLO algorithm. Control variable method is expected to be adopted in the experiment to observe the performance of each algorithm under the same weight, the same data set and the same training times, focusing on the size of FPS and mAP.

Due to the limitation of equipment and time in this project, the results could not be well presented. However, in the paper published by the author of YOLOv4, there are experiments comparing various algorithms, as shown in Fig. 12. Here, we further analyze the results.

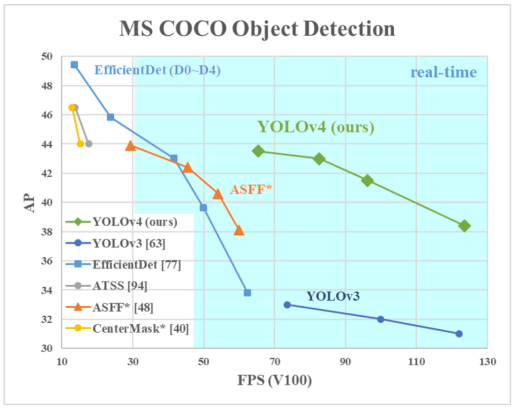


Fig. 12. The compared result

As can be seen from this image, YOLOv4 EfficientDet runs three times faster than other state-of-the-art object recognition algorithms at the time, with comparable performance, and improves AP and FPS by 10% and 12%, respectively, compared to YOLOv3.

Compared with the YOLOv4 we used, the current YOLO algorithm has become more and more mature and convenient. Now there have been YOLOv8 versions, each version has a small improvement, which is the result of continuous innovation by predecessors. Object recognition, as one of the very important components in the field of robotics and computer, will become more and more mature in the future, and the recognition accuracy will be higher and higher, and the use of human society will be greater. If you want to learn more about more advanced algorithms, please check out the updated papers in the YOLO series. And the YOLO collection doesn't stop there, with more innovations to come.

B. SLAM and Navigation

1) Task: Use simulation software gazebo to build a home environment, as shown in the figure. The cleaning robot will use the mapping algorithm to create a map based on the environment, and then use the navigation function package carried by the ROS to navigate, allowing the robot to reach the designated location.

The performance of the mapping algorithm will be evaluated by calculating the overlap between the mapping results and the predesigned map.

2) Setup: By using the open-source Gmapping algorithm and navigation package of ROS, relevant parameters are configured to achieve robot mapping and navigation in maps.

When configuring the gmapping algorithm, referring to the algorithm principles mentioned earlier, we modify and configure the following parameters, while selecting default values for the remaining parameters. The parameter 'particles

(int, default: 30)' determines the number of particles in the gmapping algorithm. Gmapping uses a particle filter algorithm, and the particles are constantly iteratively updated. Therefore, selecting an appropriate number of particles can ensure the algorithm has high speed while ensuring accuracy.The parameter 'MinimumScore (float, default: 0.0)' is the minimum matching score, which is an important parameter that determines your confidence in the laser. A higher value indicates a higher requirement for the laser matching algorithm, and the laser matching is more likely to fail and switch to using odometer data. Setting it too low can cause a lot of noise in the map.

When configuring navigation, it is mainly necessary to configure the cost map in the Move Base section. As the SLAM built map is a static map, static maps cannot be directly applied to navigation. Based on this, some auxiliary information maps need to be added to obtain auxiliary information by configuring the cost map to achieve real-time navigation obstacle avoidance function. There are two cost maps: global-Costmap and local-Costmap, the former is used for global path planning, and the latter is used for local path planning. The cost map generally has the following levels:

Static Map Layer: A static map constructed by SLAM.

Obstacle Map Layer: The obstacle information perceived by sensors in navigation.

Expansion Layer: Expand (outward) on the above two layers of the map to avoid the robot's shell colliding with obstacles.

3) Results: To evaluate the quality of the maps obtained, an analysis of the error between the generated map and the ground truth was conducted. To that end, the best fit alignment between the ground truth and the map obtained is computed, using intensity-based image registration tools. The process works as follows: Normalize two images using the image process toolbox in MATLAB, convert the size of the images to a consistent size, and obtain a grayscale image of the image. Then, compare the SSIM values (Structural Similarity Index) and RMSE values (Root Mean Square Error) of the two images to obtain the accuracy of the image. The SSIM range is [0,1], and the larger the value, the better the image quality. When two images are identical, SSIM=1. The principle of RMSE calculation is to square the difference between the real value and the predicted value, then sum and average it, and finally root it out. The smaller the RMSE value, the more similar the image is. Fig. 13 shows the results of one of the component images. Fig. 14 shows the comparison results before and after the construction of the SLAM.

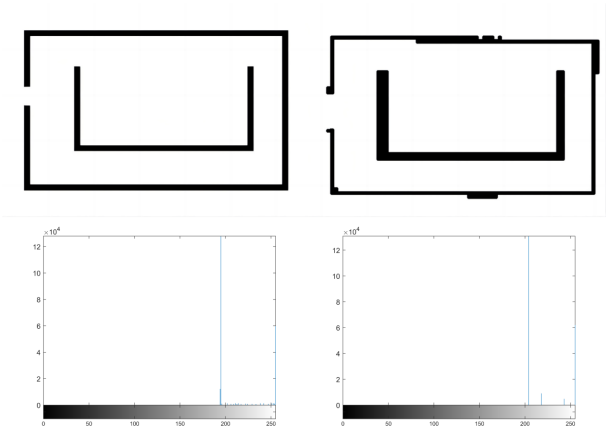
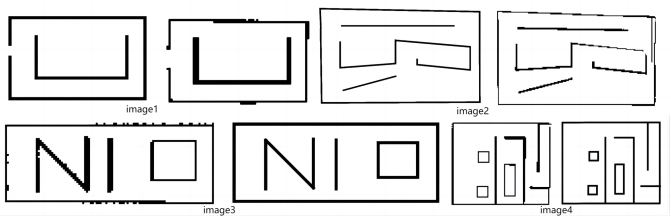


Fig. 13. The results of one of the experiments are shown here. Above the image is a comparison between the map constructed by the SLAM algorithm and the original image, while below the image are grayscale images of the two images



Simulations Experiments				
	image1	image2	image3	image4
RMSE	20.1239	91.4085	92.3797	94.5178
SSIM	0.8528	0.7035	0.6576	0.7171

Fig. 14. The upper part of the figure shows the visual comparison between the four groups of experimental Centamap and the map built by SLAM, and the lower part shows the SSIM value and RMSE value mentioned above

C. Integrated Experiment

Finally, the integration experiment is carried out to combine object recognition with robot mapping and navigation, completing the most important step of this project. The main purpose of the experiment is to verify that the robot can remove garbage in the household environment and put it in the trash can process.

The experiment is mainly divided into two parts. The first part is to verify that the integration effect of various parts of the robot can be realized in a household environment with only one garbage and one garbage can. The specific implementation process is as follows: determine the coordinates of the garbage picked up when building the world environment, write the corresponding round-trip program through the fixed-point navigation, and realize the pick-and-place of three different garbage, identify the garbage through the object, judge the distance between the robot and the garbage through the depth camera, and then execute the corresponding round-trip program. Use object recognition function to identify garbage, then obtain the location information of the object in the scene, and then execute the corresponding pick program.

The purpose of the second part of the experiment is to verify that this project can complete the process of identifying garbage in the household environment, picking up garbage, and throwing it into the trash can. Three different kinds of garbage and three different trash cans are used in this part of the experiment, aiming to carry out more research on the basis of the first part of the experiment.

The test results are shown in the Fig. 15 and Fig. 16. During

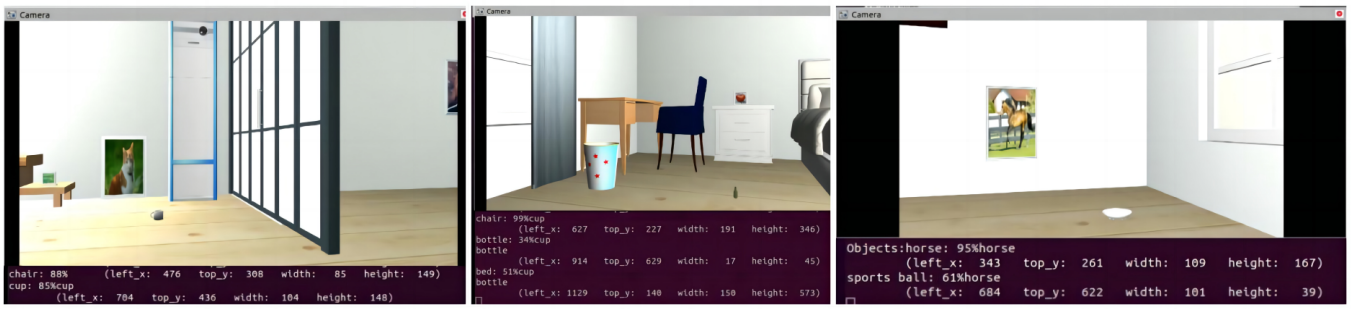


Fig. 15. This image shows the garbage we need for object recognition

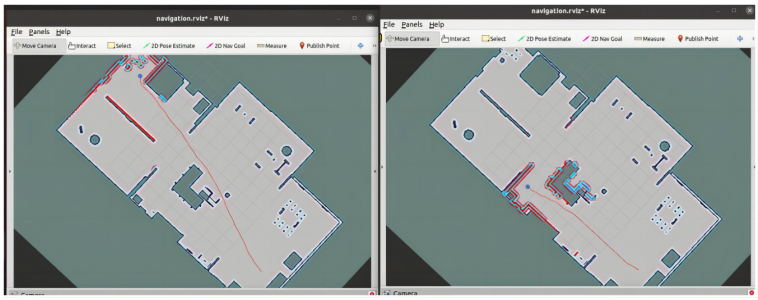


Fig. 16. the navigation path planning diagram for returning to the designated garbage storage area after picking up the garbage we have selected

the experiment, due to issues with object recognition training, the bowl garbage we designed was identified as a baseball, and the task of picking and sending it back to our designated area cannot be completed.

V. Conclusion

This project has carried out theoretical analysis, experimental verification and experimental exploration by studying the mobile robot picking up different kinds of garbage in the household environment and placing it in the corresponding trash can. Although the goal has been achieved in the end, there are still some areas for improvement.

Firstly, in terms of models and environments, models and suckers with clearer and more stable structures can be selected, and more diverse environments can be tried to meet the diversity.

Secondly, in terms of object recognition, YOLO algorithm has been developed to YOLOv8, and more advanced algorithms can be used for model training in the future, which can better improve the accuracy of object recognition. Secondly, data sets with more obvious features can be used, such as ImageNet, which not only has a large number of images, but also has more features, which can also improve the recognition effect.

Finally, in terms of navigation and mapping, more advanced mapping methods can be used to improve the mapping effect. At the same time, a more intelligent autonomous navigation system can be realized to achieve better results in completing tasks.

This project lays a good foundation for the exploration of the following tasks in this field and provides guidance for the following experiments and expansions.

Acknowledgments

Thanks to Mr. Song for the guidance of this project and to every student for their hard work.

References

[1] João Machado Santos, David Portugal, and Rui P. Rocha. An evaluation of 2d slam techniques available in robot operating system. In *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–6, 2013.

[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] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *CoRR*, abs/2004.10934, 2020.

[5] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan L. Yuille. Adversarial examples for semantic segmentation and object detection. *CoRR*, abs/1703.08603, 2017.

[6] Chien-Yao Wang, Hong-Yuan Mark Liao, I-Hau Yeh, Yueh-Hua Wu, Ping-Yang Chen, and Jun-Wei Hsieh. Cspnet: A new backbone that can enhance learning capability of CNN. *CoRR*, abs/1911.11929, 2019.

[7] Diganta Misra. Mish: A self regularized non-monotonic neural activation function. *CoRR*, abs/1908.08681, 2019.

[8] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V. Le. Dropblock: A regularization method for convolutional networks. *CoRR*, abs/1810.12890, 2018.

[9] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.