Object Grasping of Humanoid Robot Based on YOLO

Li Tian[1[0000-0002-6436-4984]], Nadia Magnenat-Thalmann[1], Daniel Thalmann[2], Zhiwen Fang[3] and Jianmin Zheng[1]

[1] Nanyang Technological University, Singapore
[2] EPFL, Switzerland
[3] A*STAR, Singapore
daniel.thalmann@epfl.ch

Abstract This paper presents a system that aims to achieve autonomous grasping for micro-controller based humanoid robots such as the Inmoov robot[1]. The system consists of a visual sensor, a central controller and a manipulator. We modify the open sourced objection detection software YOLO (You Only Look Once) v2[2] and associate it with the visual sensor to make the sensor be able to detect not only the category of the target object but also the location with the help of a depth camera. We also estimate the dimensions (i.e., the height and width) of the target based on the bounding box technique (Fig. 1). After that, we send the information to the central controller (a humanoid robot), which controls the manipulator (customised robotic hand) to grasp the object with the help of inverse kinematics theory. We conduct experiments to test our method with the Inmoov robot. The experiments show that our method is capable of detecting the object and driving the robotic hands to grasp the target object.

Keywords: Robotics, Vision, Object Detection, Motion Control, Grasping

1 Introduction

Currently, robot technology is making it possible to introduce more automation service to our daily life. Intelligent and interactive robots could assist us in home or office. One of the most important tasks is to fetch objects. However, autonomous grasping is still a not completely solved problem in robotics. In order to archive autonomous grasping, robots should have abilities to recognise the target objects and determine the location of objects. Moreover, robots need to adjust their motion trajectory based on the information of the target objects, such as location and category. Nevertheless, there are still many uncertainties that need to be further studied. Therefore, how to deal with uncertain facts and improve the success rate of grabbing is a very worthwhile problem. Generally, the uncertainties in the grasping process mainly include the uncertainties of the shape of the target object, the pose of the object, the contact point of the manipulator and the quality of the object.

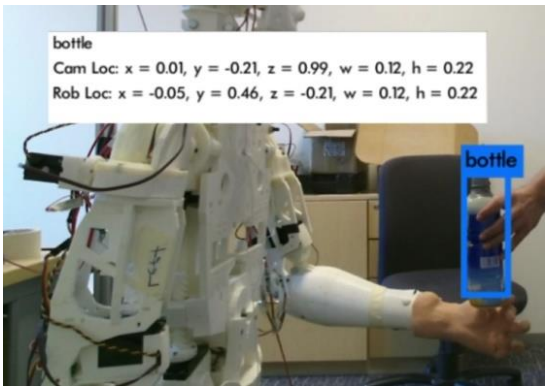

Fig. 1. Autonomous grasping with real-time object detection

To solve this problem, we establish a communication system, which includes a central controller, visual sensor and manipulator. We first collect the necessary information about the target, and then path-plan for the arm based on the position of the target. Finally, we actuate the robotic hand to grasp the target by preset grasp pose according to the categories and size of the targets. Fig. 1 shows the screenshot of our object detection software. The water bottle has been detected and highlights within the blue rectangular box. The "x,y,z" after "Cam Loc" indicate the 3D coordinate of the bottle regarding the KINECT II. And the "x,y,z" after "Roc Loc" indicate the 3D coordinate of the bottle after transformation regarding the robot's position. "w" and "h", indicate the width and height of the detected object.

The rest of the paper is organized as follows: Section 2 gives an overview of existing related work about grasping of the humanoid robot. Section 3 presents our design and method in details. Section 4 reports the experiments of evaluating the performance with the Inmoov robot.

2 Related Work

2.1 Object Detection

For robot perception, category-specific object detection (COD) is one of the most fundamental yet challenging problems in the computer vision community[3]. The object detection is actually the progress of finding predefined objects among a large number of images. One existing solution uses regions with convolutional neural network features (R-CNN) and fast versions [4, 5]. In this solution, they solve the object localisation question in two parts: 1. Generate object proposal from an image and find where it is. 2. Classify each proposal into different categories and thus recognise the object. This progress brings a tremendous amount of repeated computing, and the speed of it is slow regarding the real-time automatic grasping.

In [6] and [2], the authors compared YOLO and YOLOv2 with other existing detection frameworks and showed that their methods are faster and more accurate. Regarding the perception of the robot, the speed is fulfilled for real-time grasping.

2.2 Grasping from a humanoid robot

There are two commonly used methods to deal with the uncertain shape objects when a manipulator is used for grasping. One is to rely on other sensors, such as the tactile sensor, the force sensor, the laser sensor, to feedback more information about the object, to compensate for the shape error caused by vision sensor (camera), and finally to control the multi-degree-of-freedom machine[7]. The other is to apply the method of machine learning to the manipulator grasping[3, 8]. A large amount of data obtained through enough grasping experiments can be used as a training set of feasible grasping configurations for manipulators, and a grasping model obtained from empirical data can be obtained. In [8], they create a complete framework for the iCub robot to let it learn how to roll affordances of the object and explore handheld tools, as well as learn how to use them and finally put the learned skill in real actions.

3 The Proposed Method

3.1 Object Perception in the RGB Image

How to grasp the object remains a very difficult task. Therefore, for safe grasping in a real environment, the target objects need to be recognised and localised accurately. We define the task of grasping as fetching objects from a static place. Firstly, the system collects depth and RGB images from KINECT II. Based on the RGB images, a CNN-based object detection method named as YOLO is adapted to extract CNN features, detect objects and recognise objects. The objects are marked by rectangular boxes and labelled using the object category. Secondly, according to the rectangular boxes including the target objects, the corresponding regions are cropped from the depth image. Finally, in the depth images, object segmentation is designed to obtain the pixels belonging to the target objects, which is used to calculate the location of objects.

Next, we will introduce the details about object perception for social robot grasping. In the vision-based grasping, an interested object should be perceived in the current image at real-time speed. To facilitate real-time object perception, we adopt YOLO[2] to detect and recognise the interesting objects.

YOLO is a unified pipeline, which treats object detection as a regression problem. It can output spatially separated bounding boxes and corresponding class probabilities simultaneously. Because the pipeline of object detection is a single network, it can be optimised end-to-end directly. Moreover, YOLO can process images as a real-time speed. The pipeline of YOLO is shown in Fig. 2. It can be observed that global image features are used by YOLO to detect the objects. This framework consists of three steps: (1) resize the input image to $544×544$, (2) run a CNN-based network, and (3) output the results by the confidence of the network model.

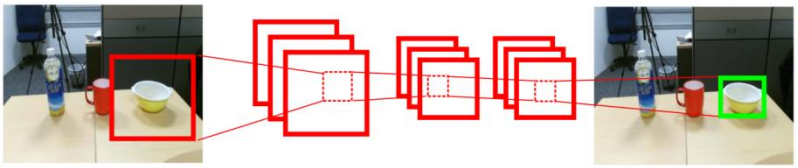

Fig. 2. The pipeline of YOLO: (1) resize the input image to 544 x 544, (2) run a CNN-based network, and (3) output the results by the confidence of network mode.

Next, we will briefly introduce YOLO. The input image is divided into a 7 x 7 grid. If a grid cell includes the centre of an object, that grid cell is used to detect that object. A bounding box and class probabilities are predicted using that grid. YOLO has 24 convolutional layers followed by two fully connected layers. It uses 1 x 1 layer followed by 3 x 3 convolutional layers and strode convolutions are used to replace the maxpooling layers.

3.2 Object segmentation from the Depth image

Having good segmentation information can be useful for grasping. It can help the social robot to estimate the location of target objects. Unlike previous methods, which only use RGB images, we take advantage of depth cues to get the pixels belonging to target objects. In order to segment the target objects from the depth image, we need to remove the background and the desk plane.

Because the length of humanoid robot arms is about 0.8m, the background information can be filtered by a fixed threshold θ as Eqn. 1.

$$RGB(p) = \begin{cases} 0, depth(p) > \theta \\ RGB(p), \ other, \end{cases}, \qquad (1)$$

where RGB(p) and depth(p) represent the RGB value and the depth value of pixel p; θ = 1000 is the threshold. For removing the background. It means that the RGB value of pixels whose depth value are larger than 1m will be set to 0. The distance-based filter can restrain the background. In order to segment objects, the depth map is also filtered as Eqn. 2.

$$depth(p) = \begin{cases} 0, depth(p) > \theta \\ depth(p), \ other, \end{cases}, \qquad (2)$$

Next, we need to get the pixels belonging to the desk. It can be observed that the plane of the desk is the biggest one because the background has been removed in the last step. In order to detect the plane, the first step is how to transform the depth image to point cloud data. IR camera intrinsic parameters, focal length and the principal point can be used to finish the transforming task[9]. The algorithm for converting the depth image to point cloud data is as Algorithm 1.

Algorithm 1: Transform the depth image to point cloud data
1: φ=1000; //Meter to MM
2: for vDepth = 1 to height

```
    for uDepth = 1 to weight
        z = depth(vDepth, uDepth)/φ
        x = (uDepth - cxd) * z / fxd
        y = (vDepth – cyd) * z / fyd
        pointcloud(vDepth,uDepth,1) = x;
        pointcloud(vDepth,uDepth,2) = y;
        pointcloud(vDepth,uDepth,3) = z;
    end
end
```

where fxd, fyd are the focal length and cxd, cyd are the principal points. After point cloud data is obtained, we adopt a RANSAC[10]-based plane fitting method to discover planes in point cloud data efficiently. The plane fitting method can get the largest set of points which fit to plane. The plane equation is defined as:

$$ax + by + cz + d = 0 \qquad (3)$$

where a, b and c are plane parameters and d is the distance from the origin to the plane. From the point cloud data, three points are selected by RANSAC which can calculate the parameters of the corresponding plane according to these points. Then a distance threshold is set to enlarge the plane. The distance threshold is set to 0.0 in our method. According to the segmentation result, the corresponding depth pixels of the desk plane are removed. Finally, we can obtain the target objects in the depth image. Based on the object detection in the RGB image and the object segmentation in the depth image, the object location (xc,yc,zc) in the camera space can be calculated by averaging the location information of pixels belonging to the target objects.

Aim to grasp the target objects, the location of the target objects should be transformed to the corresponding value in the robot space. The joint of the right arm is set as the origin of the coordinate plane in the robot space, and the formulation of the affine transformation is defined as Eqn. 4

$$\begin{pmatrix} x_r \\ y_r \\ z_r \\ 1 \end{pmatrix} = Trans(\delta x, \delta y, \delta z) Rotz(\gamma) Rotx(\beta) Roty(\alpha) \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} \qquad (4)$$

3.3 Kinematics of the arm

As Inmoov robot's arm has totally five DOFs, the kinematics calculation frame assignment is needed. After getting the position of the target object, the next task is to control the hand to approach the object. It is a typical inverse kinematics problem. It means that when a desired reachable position in the 3D space is given, the angle of each DOF can be calculated in order to move the robotic hand to the given location. It may have multiple solutions for one destiny position. The method proposed in [11] uses Eqn. 5 to calculate inverse kinematics of robot arm to get the solution with the least possible movement.

$$\Delta\theta = J^T (JJ^T + \lambda^2 I)^{-1} \vec{e} \qquad (5)$$

3.4 Robotic hand design

For a better grasping, with reference to previous work on humanoid robotic hand [12-15], we made a six DOFs 3D printable robotic hand via the processes of 3D scanning, segmentation, collision part removing and joints adding. It enables highly similar grasping when it works with artificial hand skin (Fig. 3).

Fig. 3. Our robotic hand's 3D model and the final product

4 Experiments

We have tested our robotic hand on an open-sourced humanoid robot Inmoov with our customised robotic hand. The results showed that the robot could grasp the objects autonomously in real time. Fig. 4 provides three screenshots of the robot grasp test. The procedure of our method is as follows:

1. Run our vision detection software on the controller PC. The linked KINECT II will start to detect the area in front of it.

2. When a pre-programmed target appeared, our software will send the category, location and size of the object to the humanoid robot (Fig. 1). In testing, we put a goblet in front of the robot with the human hand. Our software works fine in this case.

3. The humanoid robot will do the motion plan for the hand and fingers to grasp the target object (Fig. 4). In testing, the robot raises the left arm and hand to approach to the cup and eventually grasp the cup with wrap pose.

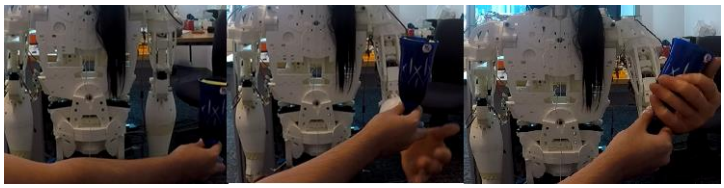

Fig. 4. Grasping test with Inmoov

Acknowledgements

This research is supported by the BeingTogether Centre, a collaboration between Nanyang Technological University (NTU) Singapore and University of North Carolina (UNC) at Chapel Hill. The BeingTogether Centre is supported by the National Research Foundation, Prime Minister's Office, Singapore under its International Research Centres in Singapore Funding Initiative.

References

1. Langevin, G., *Hand robot InMoov*. 2016.
2. Redmon, J. and A. Farhadi. *YOLO9000: better, faster, stronger*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
3. Han, J., et al., *Advanced deep-learning techniques for salient and category-specific object detection: a survey*. IEEE Signal Processing Magazine, 2018. **35**(1): p. 84-100.
4. Girshick, R. *Fast r-cnn*. in *Proceedings of the IEEE international conference on computer vision*. 2015.
5. Girshick, R., et al. *Rich feature hierarchies for accurate object detection and semantic segmentation*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.
6. Redmon, J., et al. *You only look once: Unified, real-time object detection*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
7. Roa, M.A. and R. Suárez, *Grasp quality measures: review and performance.* Autonomous robots, 2015. **38**(1): p. 65-88.
8. Tikhanoff, V., et al. *Exploring affordances and tool use on the iCub*. in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. 2013. IEEE.
9. Kurban, R., F. Skuka, and H. Bozpolat. *Plane segmentation of kinect point clouds using RANSAC*. in *The 7th international conference on information technology*. 2015.
10. Fischler, M.A. and R.C. Bolles, *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography.* Communications of the ACM, 1981. **24**(6): p. 381-395.
11. Dakarimov, S., et al., *Study on the Development and Control of Humanoid Robot Arm using MatLab/Arduino.* 유공압건설기계학회 학술대회논문집, 2018: p. 88-90.
12. Magnenat Thalmann, N., L. Tian, and F. Yao, *Nadine: A Social Robot that Can Localize Objects and Grasp Them in a Human Way*, in *Frontiers in Electronic Technologies*. 2017, Springer. p. 1-23.
13. Tian, L., et al., *The making of a 3D-printed, cable-driven, single-model, lightweight humanoid robotic hand.* Frontiers in Robotics and AI, 2017. **4**: p. 65.
14. Tian, L., et al. *A Methodology to Model and Simulate Customized Realistic Anthropomorphic Robotic Hands*. in *Proceedings of Computer Graphics International 2018*. 2018. ACM.
15. Tian, L., et al., *Nature grasping by a cable-driven under-actuated anthropomorphic robotic hand.* TELKOMNIKA, 2019. **17**(1): p. 1-7.