Vision Guided Picker

Young Il Joe
Cameron Mattson
Venkata Janakirama Pradyumna Gudluru

University of Colorado Boulder

1 Problem Space

Our goal in this project is to build a vision guided robotic (VGR) system that performs human-like tasks employing machine learning techniques. A number of autonomous robots have been developed for harvesting tasks [1]. Since it is difficult to simulate a realistic harvesting process in a robot simulator, we built a robot that picks and places oranges spawned in random places. We identified four main sub-problems for this goal. The first sub-problem is how to collect training data and annotate them for segmentation tasks. The second sub-problem deals with instance segmentation and analysis of an RGB camera image. In the third sub-problem we will address how to localize objects using a instance segmentation map and a range-finder (depth camera) in 3D space. The fourth sub-problem deals with a motion planning to reach and grab the target object while avoiding obstacles and its execution.

2 Approach

Our robot is constructed and tested in a simulated environment. The Webots robot simulator was chosen because it comes with a wide range of robots and devices, and it is easy to integrate with the robot operating system (ROS) and offers decent 3D renderings. We adapt a “UNIVERSAL ROBOTS” UR5e as a base and this robot is attached to a XY stage And a RGB camera and a range-finder are also attached to the XY stage at the same location and orientation. Also, a “ROBOTIQ” 3 Finger Gripper is attached to the end link of the UR5e robot.

Combining an image segmentation technique and an RGB-D camera such as ”Intel RealSense” or ”Microsoft Kinetic”, a robot can locate an orange in 3-dimensional space with respect to the camera.
To implement our idea of picking a fruit, we incorporated a camera view and an image segmentation technique to identify the direction to a fruit. Subsequently using a depth image. The fruit is fully localized in 3D space.

After locating an orange in 3D space, a path that moves the end effector of a robotic arm to the located orange is planned by MoveIt! [2] and executed by a a ”FollowJointTrajectory” ROS action server for UR5e robots.

2.1 Data Acquisition and Annotations

Even though public dataset for instance segmentation tasks with multiple classes of objects are publicly available such as COCO [3] and OID, we decided to create our

own dataset in the same environment where the final test was conducted. And this dataset is manually annotated using VGG Image Annotator. This decision may have limited the generalizability of our model to other environments but we hope that this choice would give a chance to learn the whole process of image segmentation task. Fig. 2. shows one of the instance segmentation training datasets taken in the Webots simulator.

2.2 Semantic Segmentation

We decided to implement the U-net architecture, because it performed well in segmentation tasks, and not as computationally expensive as some other algorithms [4]. U-net uses an contracting-expanding architecture to predict segmentation maps. In the contracting path, the network tries to summarize the most useful features to describe pixel classes by focusing on the "what". Whereas in the expanding path, the specific locations, or the "where" is determined by considering inputs from the contracting path. However, in this project, the architecture was modified to reduce the number of parameters while avoiding cropping of convolutional maps. In our architecture, we did not perform cropping before concatenating model outputs, and the output was not a grey-scale image. Alternatively, the output image only contained one channel for detecting the presence of a fruit or not. The model used "same" padding, ReLU activation functions, and dropout layer consistently. The exception was the activation function of the output layer, which was defined as sigmoid. Specifically, the threshold for whether a pixel contained an image or not was set at 0.5. The model also used binary crossentropy as the loss function and adam as the optimizer. While designing the modified U-net model, we learned that it is capable of semantic segmentation. We also attempted to cluster the predicted segmentation maps from U-net to determine how many oranges were present in each image, as well as which pixels belonged to each image.

2.3 Instance Segmentation

It has been shown that many of the instance segmentation techniques outperformed Mask R-CNN [5] in public datasets such as COCO. But we chose to train a Mask R-CNN model in the simulation for the pedagogical reason that Mask R-CNN is based on a family of computer vision techniques of Faster R-CNN [6], Fast R-CNN [7], and R-CNN [8] for the object detection task [1]. We could have likewise implemented U-net architecture, but we would have needed to adjust the algorithm to pick only one fruit at a time if multiple fruits were in the camera's field of vision.

2.4 Orange Localization

A Webots range-finder of the same FOV is placed at the same location with an RGB camera and it make the orange localization a little simpler. Two overlays on the left side in the figure above show the camera and the range-finder in the simulator.

Camera Frustum One important finding was that the name of "range-finder" may suggest it gives a distance to an object from the range-finder, but it actually gives a z component of coordinates of an object in the range-finder frame. This is effectively

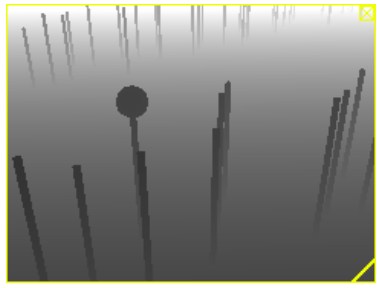

Fig. 1. A depth image generated by a Webots range-finder

under the assumption that a range-finder is operating in the portrait mode, such that w and h are the width and height of the frustum rectangle on the projection plane, which is perpendicular to the range-finder pointing direction at 1 meter away from the range-finder. Thus w and h are related with FOV as $w = 2\tan\left(\frac{\text{FOV}}{2}\right)$ and $h = w \times \text{aspect ratio}$. Then, an point at (x_c, y_c, z_c) in the range-finder frame gives the depth of z_c at depth image coordinate at

$$\left(\frac{x_c}{z_c w}, \frac{y_c}{z_c h}\right).$$

Taking the other direction, with a given depth value z of a pixel (x_i, y_i) in a depth image coordinate, the corresponding coordinates in the range-finder frame can be calculated as

$$\hat{\mathbf{p}} = (x_i z w, y_i z h, z)$$

Once it is found which pixels of a depth image correspond to an orange instance, an orange location $\hat{\mathbf{p}}$ can be estimated as above.

2.5 Motion Planning

A reinforcement learning approach employing Hindsight Experience Replay (HER) [9] technique for a 6-DOF robotic arm path planning has been tried. (After finishing the problem set 5 of this class, we realized that it is a very similar problem with a final bonus problem, except in this algorithm we have 6 degrees of freedom for a static (holonomic constraints) case or 12 degrees of freedom for a dynamic (nonholonomic constraints) case. It might have been possible to find a working policy with enough trainable parameters and simulations time. But we couldn't find a working policy in a timely manner. Thus, "RRT", one of conventional path planning technique, was employed for robotic arm path planning throuhg MoveIt!.

3 Data

3.1 Image Acquisition

Training image dataset were taken in the same simulation environment. For each episode, oranges and empty green sticks are spawned at random locations, and then

20 training images were taken. In total ten of such training episodes were conducted for a total of 200 training images. One training sample is shown in the Fig. 2.

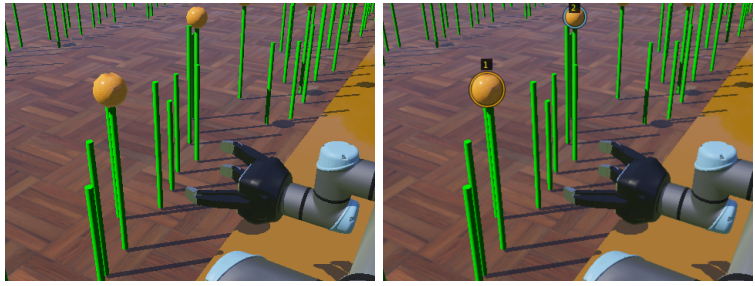

Fig. 2. The figure on the left is one of training images taken in Weobots and it contains two orange instances. The figure on the right shows two instance segmentation annotations given to this training image using the VGG image annotator.

3.2 Image Annotation

Images taken in a simulated orange field are annotated using the VGG Image Annotation tool [10]. The VIA tool supports instance segmentation tasks with different region shapes including a rectangle, circle, ellipse, and polygon. It is convenient for our purpose since most of the orange segmentation maps are circles. The annotation file is exported to a json file and these annotations are further processed in the next step

3.3 Preprocessing training data

Pytorch's implementation of Mask R-CNN requires a binary mask image for each instance in training dataset. In order to compactly represent multiple mask images in a single RGB image, an intermediate image convention was devised. In this conventions, the r-channel of our image represents the object class. For our case, there are only 2 classes. Thus, it could be zero for background or one for oranges. The g-channel of image represents the instance index. If an image has multiple instances of oranges, pixels corresponding to different oranges have different g-channel values. Additionally, the b-channel is used for our dataset. Image preprocessing was slightly different for the U-net architecture. Instead of representing mask images with the g-channel, only one channel was used to predict the absence or the presence of an orange in each picture.

The actual binary mask images were generated using the 'scikit-image' packages. A mask annotation for an instance is composed of basic geometry shapes and 'scikit-image' gives corresponding pixel coordinates for these basic geometric shapes.

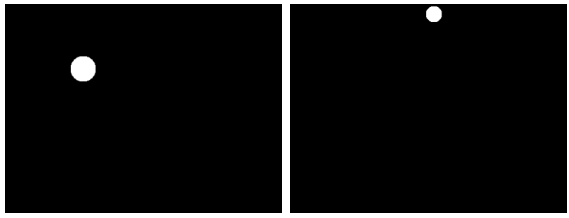

Fig. 3. Two training segmentation maps for two orange instances in 2

4 Results

4.1 Instance Segmentation

Even with a small sized training dataset, a Mask R-CNN model is successfully trained on this dataset and gives accurate enough 3D location estimations with which a robotic arm can perform pick-and-place operations for an orange.

Fig. 4 shows predictions on one of test images which also has two instances of oranges. A trained model gives two predictions with greater than 0.998 of confidence scores.

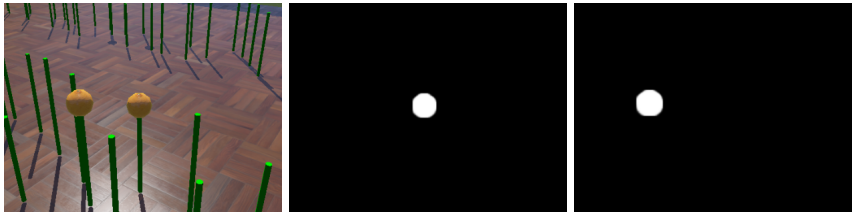

Fig. 4. The figure on the left is one of the test images that has not been used for training. The middle and right figures show two segmentation map predictions on this test image.

Precision-Recall Curve To quantify the performance of the instance segmentation model, a precision-recall curve, which is commonly used performance metric for instance segmentation tasks, is calculated at the Intersection Over Union (IoU) threshold of 0.5.

As you can see in Fig. 5 the recall never gets to 1.0 for any confidence score threshold and this indicates that the model fails to give a segmentation map IoU greater than or equal to 0.5 for some orange instances in the test dataset. Fig. 6 shows one of these failure cases. The train model fails to give any segmentation map that corresponds to any orange instance on top left of the figure.

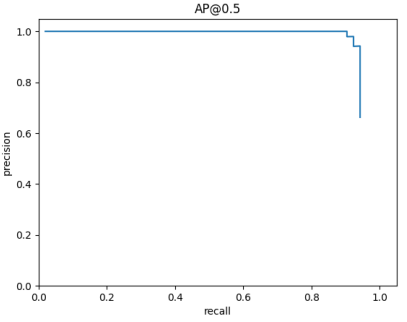

Fig. 5. The precision-recall curve on the test dataset at IoU threshold of 0.5

4.2 Semantic Segmentation

Semantic segmentation or different oranges using the U-net architecture was successful. However, in some cases, the model was not able to segment certain oranges, or falsely segmented areas that did not contain oranges. One reason this could have occurred is because the oranges in the simulation can appear visually similar to other surrounding objects due to the lack of visual detail represented in each orange. This seems likely after observing that some of the false positive segmentations were assigned to circular shadow projections on the orange flooring in the simulation, as well as other irrelevant objects. Similarly some of these misclassifications could have been the result of a deficiency in the number of training samples. The mean IoU of the modified U-net model was 0.74.

The attempted clustering algorithms were unable to cluster very many pixels of fruit segmentation maps even when only one fruit segmentation map was present for an image.

4.3 Complete Robotic Operation

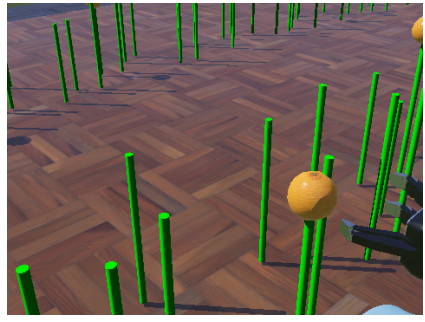

Fig. 6. Even to human eyes, it may take some time to locate all of 3 orange in this figure. One of the oranges is located in the top left area and partially blocked by a green cylinder.

With each cycle we first detect an orange of interest, move the XY stage for the second detection, detect the orange again, localize the orange, and then plan and execute the pick-and-place operation. Repeating these cycles, oranges are placed on one side of the field by the robotic arm. Fig. 7 shows the pick-and-place operation just after an orange is picked up by the end effector.

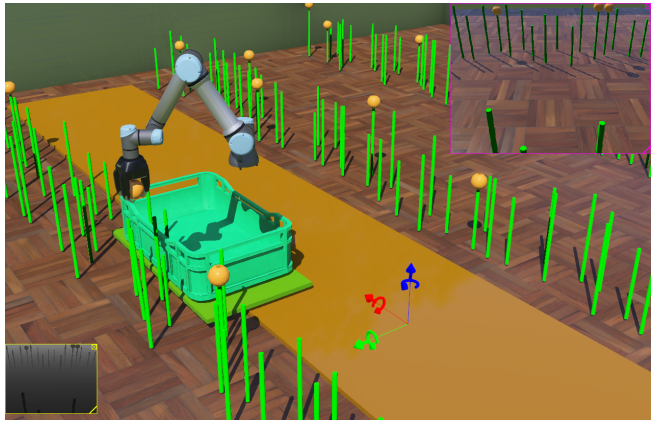

Fig. 7. A pick-and-place operation.

5 Discussion

Even though our simulated environment for an orange field is not very realistic, we successfully build an autonomous orange picker by combining image segmentation algorithms, range-finder, RRT motion planning, and ROS "FollowJointTrajectoryAction" action server for UR5e. This demonstrates that an image segmentation technique can be applied to vision guided robots to perform autonomous operations. Since our ultimate goal is picking oranges, few false negatives of instance segmentation is not critical. As long as a oranges can be detected at a certain angle and position, the robot can approach the fruit and pick it up. To measure the overall performance of an autonomous orange picker, appropriate performance measures or benchmarks need to be devised.

Using both the U-net and clustering algorithms was not a viable alternative to instance segmentation algorithms, since pixels of adjacent object instances may not be separable using current clustering techniques. However, using this approach for disjoint segmentation tasks could be used a novel approach.

We see many possible improvements over the current implementation. Listed below are a few such improvements:

1. In the current path planning implementation, the XY stage movement and the the robotic arm movements are sequentially executed. It's designed in this way since it is simple to implement but it obviously gives a sub-optimal motion path.
2. A camera is attached to the XY stage and orange localization is executed once per each orange. But we believe that more appropriate position would be near the end-effector. And this would gives a series of feedback of the locations of an orange as the robotic arm approaches the orange. Consequently, it would make a more reliable pick-and-place operation.
3. With a limited training, the motion control policy could not be successfully constructed using the reinforcement learning technique (HER). But we believe with enough training and some human guidance for training sequences, a reinforcement learning approach would give a efficient motion control.
4. The current implementation of orange localization from a point cloud is handcrafted. But we believe that this task is also machine-learnable. With enough training, it would give a more robust orange localization algorithm especially given partially blocked oranges.
5. Grasping is possibly a complicated task. In this project, the end-effector only needs to deal with oranges and manual grasping operations to confer a good performance. In order to deal with objects with different shape and size, we believe that machine learning approaches can be applied to calculate grasping operations for a variety of objects. [11]

References

1. Ganesh, P., Volle, K., Burks, T., Mehta, S.: Deep orange: Mask r-cnn based orange detection and segmentation. IFAC-PapersOnLine **52**(30) (2019) 70–75 6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture AGRICONTROL 2019.
2. https://moveit.ros.org/
3. https://cocodataset.org/home
4. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention, Springer (2015) 234–241
5. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn (2017)
6. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R., eds.: Advances in Neural Information Processing Systems. Volume 28., Curran Associates, Inc. (2015)
7. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). (December 2015)
8. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (June 2014)
9. Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., Zaremba, W.: Hindsight experience replay. In Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., eds.: Advances in Neural Information Processing Systems. Volume 30., Curran Associates, Inc. (2017)
10. https://www.robots.ox.ac.uk/ vgg/software/via/
11. Du, G., Wang, K., Lian, S., Zhao, K.: Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review. Artificial Intelligence Review **54**(3) (aug 2020) 1677–1734