HSR Grasping

Orlando Fraser

Summer 2020

1 Introduction

The aim of the project was to design an algorithm that will enable the HSR robot to execute grasps on fine objects such as pens or cutlery. This algorithm could then be used for a range of tasks that the robot needs to carry out. The novel contribution of this work is to use images from the RGB hand camera as input to the grasping algorithm rather than the main depth camera on the robot's head. The benefit of the hand camera is its ability to get closer to the desired objects and is less prone to obstructions to its field of view.

2 Background and Literature Review

the research area of robotic grasping has seen a range of work in recent years. Most approaches use depth images as input to the algorithm, using this to predict an optimal grasp of the object in the field of view. This grasp is defined by a vector that contains a position, an orientation, and a gripper width. The grasp is in 2D image space, so it then converted to 3D world space using a known set of transforms. The grasp vector can then be used to find an error between the current position of the gripper/hand and the desired position (the grasp vector itself). This error can then be minimized in order to successfully execute the grasp.

RGB images could be used as input to a grasping algorithm and they would (in theory!) be able to generate a grasp from the data. However depth data is required to convert this grasp from 2D image coordinates to 3D world coordinates. For this reason, you may as well use depth for the input and so, to our knowledge, there are no current methods that use RGB alone at this time.

We therefore can separate, the grasping algorithm into three distinct sections: depth estimation from RGB, grasp prediction from the depth map, and finally, the controller that will provide the necessary instructions to execute the grasp in a efficient manner. We elaborate on the design choices made for each of these sections below.

2.1 Depth Estimation

The hand camera is just RGB, however we need a depth map in order to generate a grasp. The problem of estimating depth from RGB images is an active research area called Monocular depth estimation. This enables a rough depth map to be produced from an RGB image, this depth map can then be used with the grasping algorithm.

The majority of current algorithms in this area use a CNN based approach trained with a dataset made up of RGB images along with the ground truth depth maps. We tested some popular algorithms but they produced very poor quality depth maps on our data. This was because most standard methods were geared towards wide angle outdoor scenes (eg for self driving cars) and therefore they didn't generalize well to our application, which consisted of more close up indoor images.

The recent approach from Ranftl et al [2] was designed to address the poor generalizabilty of previous methods by utilizing data from a range of datasets.

2.2 Grasp Prediction

The 2018 paper from Morrison et al. [1] detailed an algorithm that used convolutional neural networks to predict grasps (GG-CNN). It took took a 300x300 depth image as input and predicts a grasp for each pixel in this input, this output is called the grasp map. The grasp map consists of a quality, angle, and width map, each of the same dimension as the input. One can then find the optimal grasp by find the

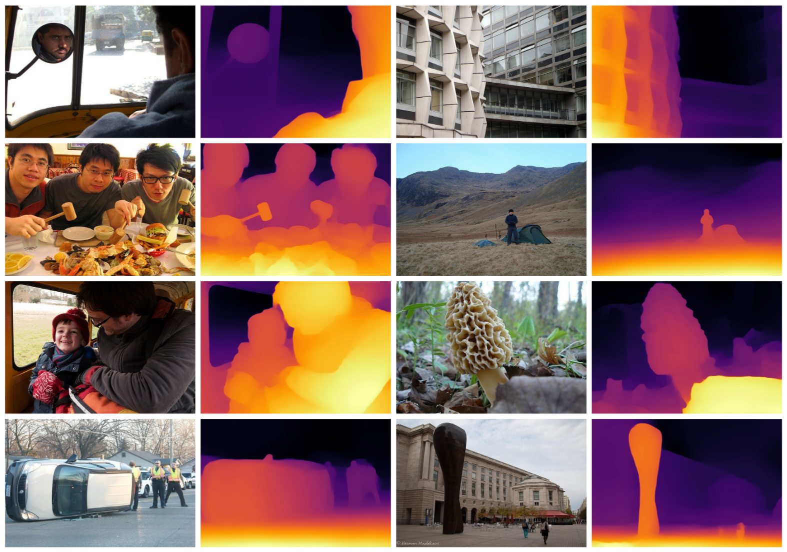

Figure 1: The depth estimation algorithm tested on a range of different scenarios. Exhibits robust performance across the range of scales and lighting conditions.

max point of the quality map.

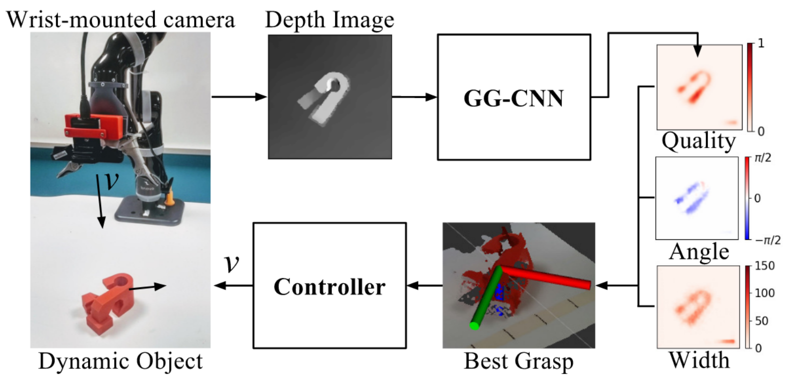

Figure 2: Pipeline showing the GG-CNN algorithm.

2.3 Control

Both open loop and closed loop approaches can be used for the controller. In open loop, the grasp is calculated once and the error is minimized in a single step. Whereas in a closed loop approach, after the grasp is calculated, a small step is taken in a direction that minimizes the error, after which the grasp is recalculated from the new position and the process repeats until convergence.

The benefit of open loop systems is that the grasping algorithm only needs to be run once per grasp, therefore reducing the total amount of computation required. However, the lack of feedback means that they have no way of correcting the movement if an error arises in any part of the system. Closed loop systems have feedback which enables errors to be corrected quickly, leading to a much more robust algorithm. However they require a algorithm that can run at a speed of at least 30Hz or else the movement will be too slow to be viable.

3 Testing with the HSR simuation

The algorithm was tested with the HSR simulation on Gazebo. The HSR python API was used in conjunction with a Jupyter notebook to send and receive commands/data from the simulation. This enabled us to run the various python algorithms on the simulation in a simple way.

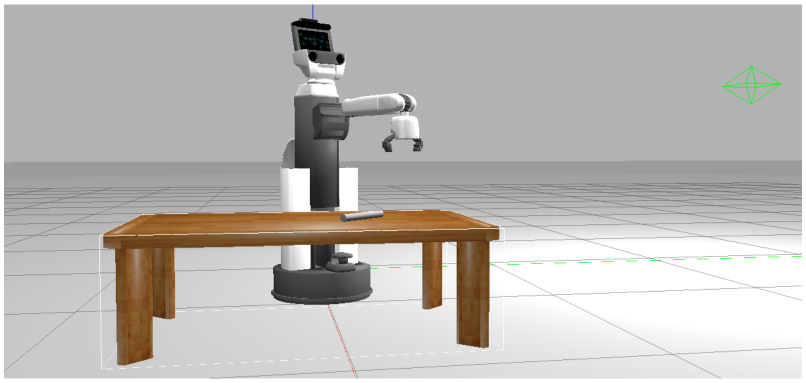

Figure 3: Simple Gazebo set up that was used to test the algorithm.

Due to constraints imposed by the COVID-19 pandemic, we were unable to test the system on the real HSR robot. This is left for future work.

References

[1] Morrison, D., Corke, P. and Leitner, J., 2018. Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. arXiv preprint arXiv:1804.05172.

[2] Lasinger, K., Ranftl, R., Schindler, K. and Koltun, V., 2019. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. arXiv preprint arXiv:1907.01341.