Learning to Detect Grasp Affordances of 3D Objects using Deep Convolutional Neural Networks

Yikun Li, Mario Rios-Munoz, Lambert Schomaker, S. Hamidreza Kasaei

***Abstract*— Grasp affordances detection is one of the challenging tasks in robotics because it must predict the grasp configuration for the object of interest in real-time to enable the robot to interact with the environment. In this paper, we present a new deep learning approach to detect object affordances for a given 3D object. The method trains a Convolutional Neural Network (CNN) to learn a set of grasping features from RGB-D images. We named our approach *Res-U-Net* since the architecture of the network is designed based on U-Net structure and residual network-styled blocks. It devised to be robust and efficient to compute and use. A set of preliminary experiments has been performed to assess the performance of the proposed approach regarding grasp success rate on simulated robotic scenarios. Experiments validate the promising performance of the proposed architecture on a subset of ShapeNet dataset and simulated robot scenarios.**

I. Introduction

Traditional object grasping approaches have been used in service robots, factories assembly lines, and many other areas widely. In such domains, robots broadly work in tightly controlled conditions to perform object manipulation tasks. Nowadays, robots are entering human-centric environments. In such places, generating stable grasp pose configuration for the object of interest is a challenging task due to the high demand for accurate and real-time response under changing and unpredictable environmental conditions [1]. In human-centric environments, an object may have many affordances, where each one can be used to accomplish a specific task. As an example, consider a robotic cutting task using a knife. The knife has two affordances parts: the handle and the blade. The blade is used to cut through material, and the handle is used for grasping the knife. Therefore, the robot must be able to identify all object affordances and choose the right one to plan the grasp and complete the task appropriately.

In this paper, we approach the problem of learning deep affordance features for 3D objects using a novel deep Convolutional Neural Network and RGB-D data. Our goal is to detect robust object affordances from rich deep features and show that the robot can successfully perform grasp actions using the extracted features in the environment. Towards this goal, we propose a novel neural network architecture namely Res-U-Net designed to be robust and efficient to compute and use. Besides, we propose a grasping approach to use the generated affordances and produce grasping trajectories for a parallel-plate robotic gripper. We carry out experiments to evaluate the performance of the proposed approaches in a virtual environment. Fig. 1 shows ten examples of our approach.

The authors are with the Faculty of Science and Engineering, Artificial Intelligence and Computer Science, University of Groningen, 9700 AB Groningen, The Netherlands. {y.li.76, m.rios.munoz}@student.rug.nl, {hamidreza.kasaei,l.r.b.schomaker}@rug.nl

The remainder of this paper is organized as follows. In the next section, related work is discussed. Three CNN-based grasp affordances detection approaches are then introduced in section III. The detailed methodologies of grasping approach are presented in section IV, then we apply the neural network with the proposed grasp approach in a simulation environment and explain experimental evaluation in section V. Finally, conclusions are presented, and future directions are discussed in section VI.

II. Related Work

Object grasping has been under investigation for a long time in robotics. Although an exhaustive survey is beyond the scope of this paper, we will review a few recent efforts.

Herzog et al. [2] assumed the similarly shaped objects could be grasped similarly and introduced a novel grasp selection algorithm which can generate object grasp poses based on previously recorded grasps. Vahrenkamp et al. [3] shown a system that can decompose novel object models by shape and local volumetric information, and label them with semantic information, then plan the corresponding grasps. Song et al. [4] developed a framework for estimating grasp affordances from 2D images (texture and object category are taken into consideration). Kopicki et al. [5] presented a method for one-shot learning of dexterous grasps and grasp generation for novel objects. They trained five basic grasps at the beginning and grasped new objects by generating grasp candidates with contract model and hand-configuration model. Kasaei et al. [6] introduced interactive open-ended learning approach to recognize multiple objects and their grasp affordances. When grasping a new object, they computed the dissimilarity between the new object and known objects and found the most similar object. Then, they try to adopt corresponding grasp configuration. If the dissimilarity is larger than the preset threshold, a new class will be created and learned. Kasaei et al. [7] proposed a data-driven grasp approach to grasp the household object by using top and side grasp strategies. It has been reported that they cannot be applied to grasp challenging objects, e.g., objects that should be grasped by their handle or grasped vertically as for instance a plate [8].

Over the past few years, extraordinary progress has been made in robotic application with the emergence of deep learning approaches. Nguyen et al. [9] researched on detecting grasp affordances using RGB-D images and got

Fig. 1: Ten examples of affordance detection results.

satisfactory results. They trained a deep Convolutional Neural Network to learn the in-depth features (object grasp affordances) from the camera images and which is proved to outperform the other state-of-the-art methods. Qi et al. [10] studied deep learning on point sets, and they proved the deep neural network can efficiently and robustly learn from point set features.

III. Affordance Detection

The input to our CNN is a point cloud of an object, which is extracted from a 3D scene using object detection algorithms such as [11], [12]. The point cloud of the object is then fed into a CNN to detect an appropriate grasp affordance of the object. Our approach consists of two main processes, including data representation of 3D objects and training of CNN on represented data. We use three types of neural networks to learn the object affordances features from 3D objects. In the following subsections, we describe the detail of each process.

A. Data Representation

A point cloud of an object is represented as a set of points, $p_i : i \in \{1, \dots n\}$, where each point is described by their 3D coordinates $[x, y, z]$ and RGB information. In this work, we only used geometric information of the object. Therefore, the input and output data type is point cloud which stored in a three dimensions array. Towards this end, we first represent an object as a volumetric grid and then use the obtained representation as the input to the CNN with 3D filter banks. In this work, considering the computational power limit, we use a fixed occupancy grid of size $32 \times 32 \times 32$ voxels as the input of networks.

B. Proposed Architecture

In this section, we propose a new network architecture to tackle the problem of grasp affordances detection for 3D objects using a volumetric grid representation and 3D deep CNN. In particular, our approach is a combination of U-Net and residual network [13]. To make our contribution more transparent, we compare our approach with the architecture of the encoder-decoder network and U-Net and highlight the similarities and differences [14]. All the networks contain two essential parts: one is the encoder network, and the other is the decoder network.

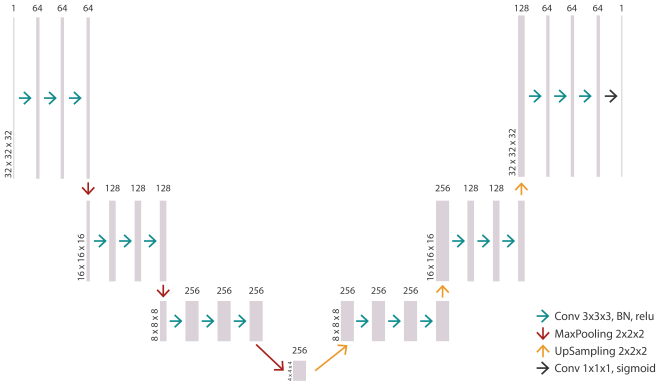

Fig. 2: Structure of the encoder-decoder network. Each grey box stands for a multi-channel feature map. The number of channels is shown on the top of the feature map box. The shape of each feature map is denoted at the lower left of the box. The different color arrows represent various operations shown in the legend.

The architecture of the encoder-decoder network is depicted in Fig. 2. This architecture is the lightest one among the selected architectures in terms of the number of parameters and computation, making the network easier and faster to learn. The encoder part of this network has nine 3D convolutional layers (all of them are $3 \times 3 \times 3$), and each of them is followed by batch normalization and ReLU layer. At the end of each encoder layer, there is a 3D max-polling layer of $2 \times 2 \times 2$ to produce a dense feature map. Each encode layer is corresponding to a decoder layer. It also has nine 3D convolutional layers. The difference is that instead of having 3D max-polling layers, at the beginning of each layer, an up-sampling layer is utilized to produce a higher resolution of the feature map. Besides, a $1 \times 1 \times 1$ convolutional layer and a sigmoid layer is attached after the final decoder to reduce the multi-channels to 1.

The architecture of U-Net [14] is shown in Fig. 4. The basic structure of the U-Net and the described encoder-decoder network are almost the same. The main difference is that, in U-Net architecture, the dense feature map is first copied from the end of each encoder layer to the beginning of each decoder layer, and then the copied layer and the up-sampled layer are concatenated.

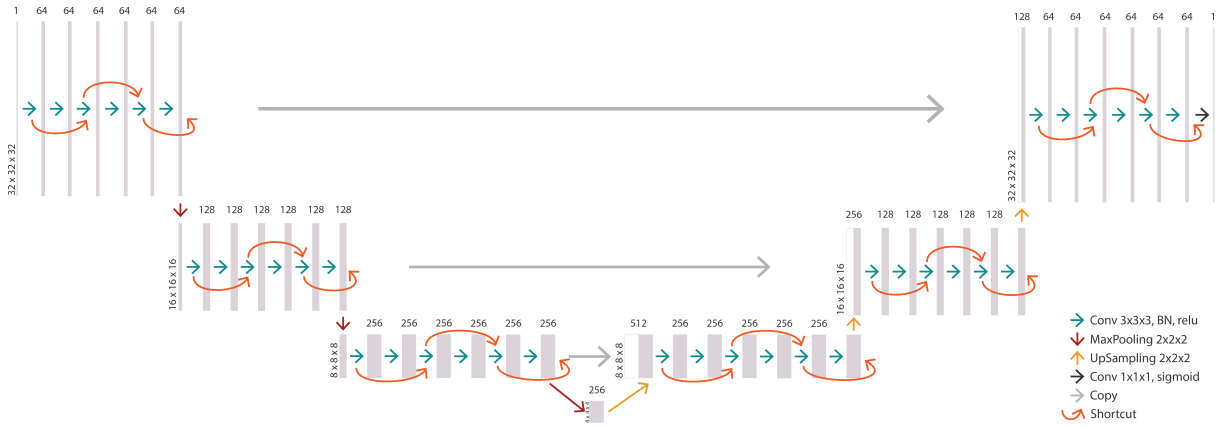

Fig. 3: Structure of the proposed Res-U-Net: compared to the U-Net, we replace the residual blocks with 3D convolutional layers and skipping over layers. This skipping over layers effectively simplifies the network and speeds learning by reducing the impact of vanishing gradients.

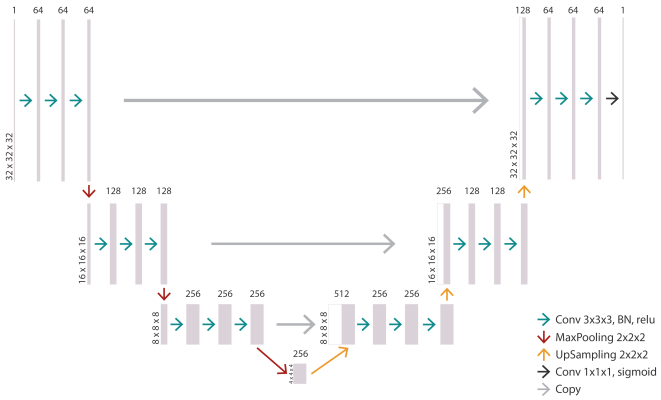

Fig. 4: Structure of the U-network: compared to the encoder-decoder network, the last feature map of each layer in the encoder part is copied and concatenated to the first feature map of the same layer in the decoder part.

The architecture of our approach is illustrated in Fig. 3. We call this network Res-U-Net. To retain more information from the input layer and dig more features, inspired by the residual network [13], we come up with this new network architecture. Compared to the U-Net, we replace the residual blocks with 3D convolutional layers and skipping over layers. The main motivation is to avoid the problem of vanishing gradients, by reusing activations from a previous layer until the adjacent layer learns its weights. Benefiting from the residual blocks, the network can go deeper since it simplifies the network, using fewer layers in the initial training stages.

IV. Grasp Approach

As we mentioned in the previous section, we assume the given object is laying on a surface, e.g., a table. The object is then extracted from the scene and fed to the Res-U-Net as shown in Fig. 5 (*a-c*) After detecting the graspable area of the given object, the point cloud of the object is further processed to determine grasp points and an appropriate grasp configuration (i.e., grasp point and end-effector positions and orientations) for each grasp point. In particular, the detected affordance part of the object is first segmented into three clusters using the K-means algorithm. The centroid of each cluster indicates one grasp candidate (Fig. 5 (*d*)) and is considered as one side of the approaching path. We create a pipeline for each grasp candidate and process the object further to define the other side of the approaching path. Inside each pipeline, we generate a Fibonacci sphere with setting the center of the sphere at the grasp candidate and then randomly select N points on the sphere. We then define N linear approaching paths by calculating lines using selected points and the grasp candidate point (i.e., the center of the sphere). In our current setup, N has been set to 256 points which are shown by red lines Fig. 5. In this study, we use a set of procedures to define the best approaching path:

- **Removing the approaching paths which are started from the under-table:** by considering the table information, we remove infeasible approaching paths, i.e., those paths that their start point is under the table (*see the second image in each pipeline*).
- **Computing the main axis of the affordance part:** Principal Component Analysis (PCA) is used to compute the axes of minimum and maximum variance in the affordance part. The maximum variance axis is considered as the main-axis (shown by a green line in the third image of each pipeline).
- **Calculating a score for each approaching path:** the following equation is used to calculate a score for each approaching path:

$$score = 2\frac{\pi - a}{\pi}\sum_{i=1}^{n} min(1, \frac{1}{d^2 + \epsilon}) \qquad (1)$$

where n represents the number of points of the object, d stands for the distance between the specific approaching path and one of the points in a point cloud model, ϵ is equal to 0.01, and a is the angle between approaching path line and the main axis of the affordance part, ranging from 0 to $\frac{\pi}{2}$. Since [15] has shown that humans

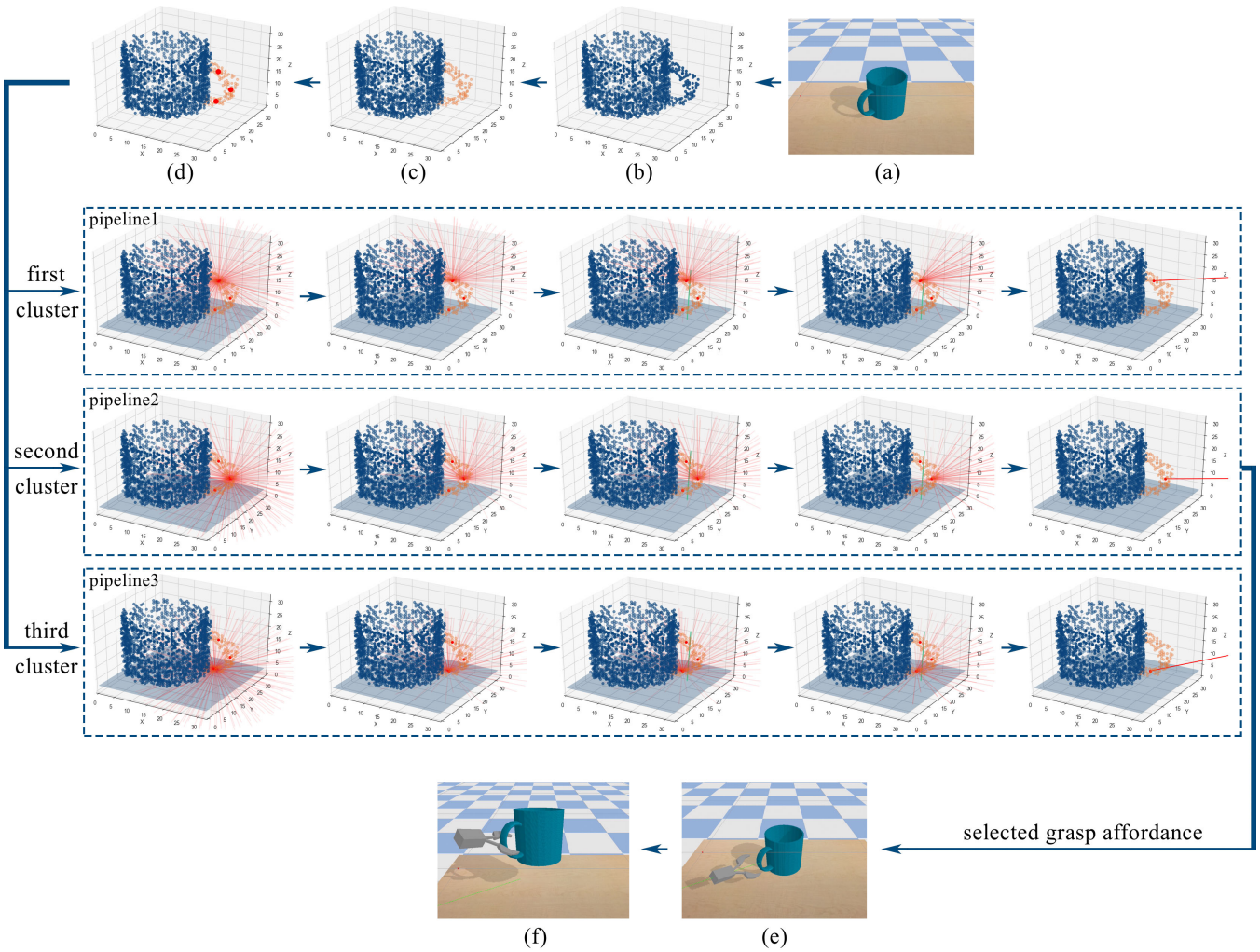

Fig. 5: An illustrative example of detecting affordance for a *Mug* object: (*a*) a *Mug* object in our simulation environment; (*b*) point cloud of the object; (*c*) feeding the point cloud to Res-U-Net for detecting the graspable part of object (highlighted by orange color); (*d*) the identified graspable area is then segmented into three clusters using the K-means algorithm. The centroid of each cluster is considered as a graspable point. Then, the point cloud of the object is further processed in three pipelines to find out an appropriate grasp configuration (end-effector positions and orientations) for each graspable point. In particular, inside each pipeline, a set of approaching paths is first generated based on the Fibonacci sphere (shown by red lines) and the table plane information (shown by a dark blue plane); we then eliminate those paths that go through the table plane. Afterward, we find the principal axis of the graspable part by performing PCA analysis (the green line shows the main axis), which is used to define the goodness of each approaching path. The best approaching path is finally detected and (*e*) used to perform grasping; (*f*) this snapshot shows a successful example of grasp execution.

tend to grasp object orthogonally to the principal axis, we then calculate $(2 * \frac{\pi - a}{\pi})$ in the formula to reduce the score when the path is orthogonal to the principal axis. The lower score means the distances between the approaching path to all points of the objects are farther. Therefore, the path with the lowest score is selected as a final approaching path for each grasp point candidate. The approaching paths with scores' influence are shown as the fourth image in each pipeline. It is visible that all paths with deeper color represent proper approaching paths. Finally, the best approaching path is selected as the approaching path for the given grasp point (*last figure in each pipeline*).

After calculating a proper approaching path, we instruct the robot to follow the path. Towards this end, we first transform the approaching path from object frame to world frame and then dispatch the planned trajectory to the robot to be executed (Fig. 5 (*e* and *f*). It is worth to mention, in some situation it is possible that the fingers of the gripper get in contact with the table (which stops the gripper from moving forward). To handle this point, we do slight roll rotation on the gripper to find a better angle between gripper and table to keep gripper moving forward. An illustrative example of the proposed grasp affordance detection is depicted in Fig. 5.

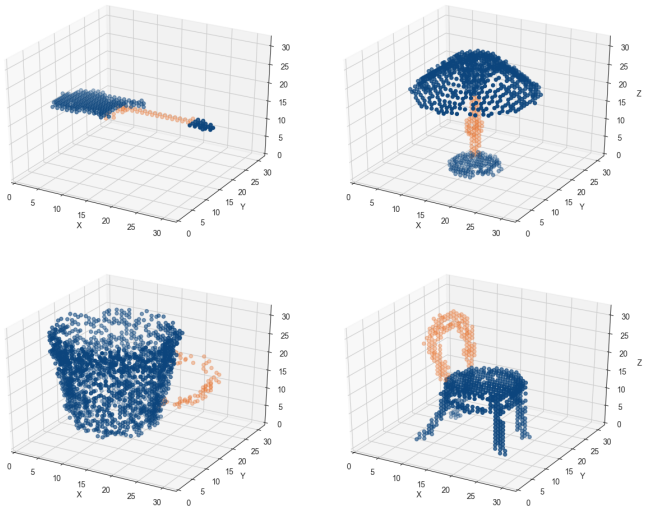

Fig. 6: Example of labeled point clouds: point cloud of the object is shown by dark blue and labeled affordance part of each object is highlighted by orange color.

V. Experiments and Results

A set of experiments was carried out to evaluate the proposed approach. In this section, we first describe our experimental setup and then discuss the obtained results.

A. Dataset

In these experiments, we mainly used ShapeNet dataset [16], which contains more than 51,300 different 3D models. In particular, our five object categories include *Mug, Chair, Knife, Guitar*, and *Lamp*. We randomly selected 500 object models from each category, and then convert these 3D models into complete point clouds with the pyntcloud package, then shift and resize the point clouds data and save them in a $32 \times 32 \times 32$ array as the input size of networks.

Since there are no existing similar researches done before, we manually labeled an affordance part for each object to provide ground truth data. A set of examples of labeled an affordance part for different objects is depicted in Fig. 6 (affordance parts are highlighted by orange color). It should be noted that we augment the dataset with by rotating the point clouds along the z-axis for 90, 180 and 270 degrees and flip the point clouds vertically and horizontally from the top view to augment the training and validation data. We obtain 2580 training and 588 validation data for evaluation.

B. Training

We start by explaining the training setup. All the proposed networks are trained from scratch through RMSprop optimizer with the ρ setting to 0.9. We initially set the learning rate to 0.001. If the validation loss does not decrease in 5 epochs, the learning rate is decayed by multiplying the square root of 0.1 until it reaches the minimum learning rate of 0.5×10^{-6}. The binary cross-entropy loss is utilized in training and the batch size is set to 16. We mainly use Python and Keras library in this study. The training process takes around two days on our NVIDIA Tesla K40m GPU, depending on the complexity of the network.

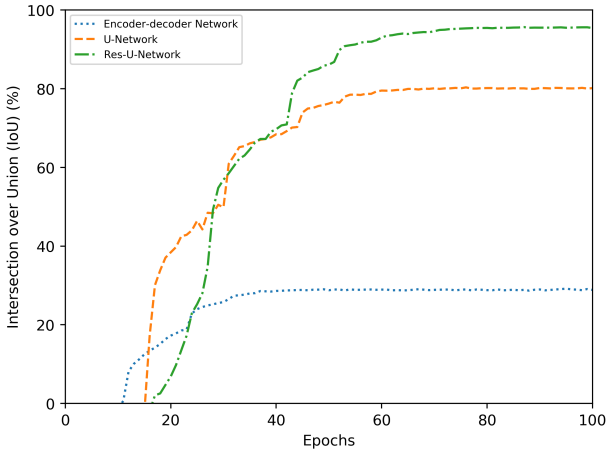

Fig. 7: Intersection over Union (IoU) on the training dataset over training epochs.

C. Affordance Detection Results

Figure 7 and Figure 8 show the results of affordance detection by three neural networks on our dataset. By comparing all the experiments, it is visible that the encoder-decoder network performs much worse than the other two counterparts. In particular, the final Intersection over Union (IoU) of the encoder-decoder network was 28.9% and 22.3% on training and validation data receptively. The U-network performs much better than the encoder-decoder network. Its final IoU is 80.1% and 71.4% on training and validation dataset, receptively. Our approach, Res-U-Net, clearly outperformed the others by a large margin. The final IoU of Res-U-Net was 95.5% and 77.6% on training and validation dataset respectively. Particularly, in the case of training, it was 15.4 percentage points (p.p.) better than U-Net and 66.6 p.p. better than the encoder-decoder network, in the case of

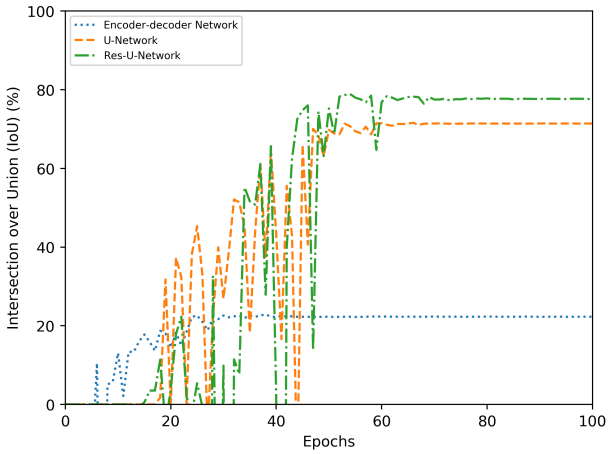

Fig. 8: Intersection over Union (IoU) on the validation dataset over training epochs.

TABLE I: Grasp success rate

Category	Success rate (%)	Success / Total
Mug	80	8 / 10
Chair	100	10 / 10
Knife	90	9 / 10
Guitar	80	8 / 10
Lamp	80	8 / 10
Average	86	43 / 50

validation, it was 6.2 p.p., and 55.3 p.p. better than U-Net and encoder-decoder network respectively.

D. Grasping Results

We empirically evaluate our grasp methodology using a simulated robot. In particular, we build a simulation environment to verify the capability of our grasp approach. The simulation is developed based on the Bullet physics engine. We only consider the end-effector pose $(x, y, z, roll, pitch, yaw)$ to simplify the complexity and concentrate on evaluating the proposed approach.

We design a grasping scenario that the simulated robot first grasps the object and then picks it up to a certain height to see if the object slips due to bad grasp or not. A particular grasp was considered a success if the robot is able to complete the task. In this experiment, we randomly selected 10 different objects for each of the five mentioned categories. In each experiment, we randomly place the object on the table region and also rotate it along the z-axis. It is worth to mention that all test objects were not used for training the neural networks. Table I shows the experimental results of grasping success rate. Figure 1 shows the grasp detection results of ten example objects. A video of this experiment is available online at `http://youtu.be/5_yAJCc8owo`

VI. Conclusion and Future Work

In this paper, a deep convolutional neural network structure for object affordance detection and an approach for grasping are introduced. We demonstrate that our Res-U-Net outperforms the other two networks in affordance detection. Moreover, we build a simulation environment and carry out experiments showing that the gripper can successfully perform grasping tasks using the proposed methods.

However, our grasping approach was trained to grasp only five object categories; we try to see the performance of our grasp approach by a set of completely unknown objects. As shown in Fig. 9, The robot could grasp and completed the grasping scenario. This observation showed that our network has already had some capacities for extracting some common basic grasping features. In particular, we believe that the affordance parts of new objects that are similar to known ones (i.e., they are familiar) can be grasped in a similar way. In the future, we will try to train the network using more object categories and evaluate its generalization power using a set of unknown objects.

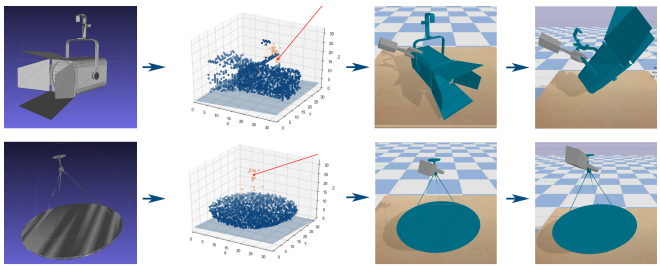

Fig. 9: Examples of grasping unknown objects by recognizing the appropriate affordance part and approaching path.

References

[1] J. J. Gibson, *The ecological approach to visual perception: classic edition*. Psychology Press, 2014.

[2] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, T. Asfour, and S. Schaal, "Template-based learning of grasp selection," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 2379–2384.

[3] N. Vahrenkamp, L. Westkamp, N. Yamanobe, E. E. Aksoy, and T. Asfour, "Part-based grasp planning for familiar objects," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 919–925.

[4] H. O. Song, M. Fritz, D. Goehring, and T. Darrell, "Learning to detect visual grasp affordance," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 798–809, 2015.

[5] M. Kopicki, R. Detry, M. Adjigble, R. Stolkin, A. Leonardis, and J. L. Wyatt, "One-shot learning and generation of dexterous grasps for novel objects," *The International Journal of Robotics Research*, vol. 35, no. 8, pp. 959–976, 2016.

[6] S. H. Kasaei, M. Oliveira, G. H. Lim, L. S. Lopes, and A. M. Tomé, "Towards lifelong assistive robotics: A tight coupling between object perception and manipulation," *Neurocomputing*, vol. 291, pp. 151–166, 2018.

[7] S. H. Kasaei, N. Shafii, L. S. Lopes, and A. M. Tomé, "Object learning and grasping capabilities for robotic home assistants," in *Robot World Cup*. Springer, 2016, pp. 279–293.

[8] N. Shafii, S. H. Kasaei, and L. S. Lopes, "Learning to grasp familiar objects using object view recognition and template matching," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2895–2900.

[9] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, "Detecting object affordances with convolutional neural networks," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2765–2770.

[10] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 652–660.

[11] S. H. Kasaei, J. Sock, L. S. Lopes, A. M. Tomé, and T.-K. Kim, "Perceiving, learning, and recognizing 3d objects: An approach to cognitive service robots," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[12] J. Sock, S. Hamidreza Kasaei, L. Seabra Lopes, and T.-K. Kim, "Multi-view 6d object pose estimation and camera motion planning using rgbd images," in *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[15] R. Balasubramanian, L. Xu, P. D. Brook, J. R. Smith, and Y. Matsuoka, "Human-guided grasp measures improve grasp robustness on physical robot," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 2294–2301.

[16] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.