*Transdisciplinary Engineering Methods for Social Innovation of Industry 4.0*
*M. Peruzzini et al. (Eds.)*


*doi:10.3233/978-1-61499-898-3-281*

# Deep Learning-Based Method for Vision-Guided Robotic Grasping of Unknown Objects

Luca BERGAMINI, Mario SPOSATO, Margherita PERUZZINI[1],
Roberto VEZZANI and Marcello PELLICCIARI
*University of Modena and Reggio Emilia, Italy*

**Abstract.** Collaborative robots must operate safely and efficiently in ever-changing unstructured environments, grasping and manipulating many different objects. Artificial vision has proved to be collaborative robots' ideal sensing technology and it is widely used for identifying the objects to manipulate and for detecting their optimal grasping. One of the main drawbacks of state of the art robotic vision systems is the long training needed for teaching the identification and optimal grasps of each object, which leads to a strong reduction of the robot productivity and overall operating flexibility. To overcome such limit, we propose an engineering method, based on deep learning techniques, for the detection of the robotic grasps of unknown objects in an unstructured environment, which should enable collaborative robots to autonomously generate grasping strategies without the need of training and programming. A novel loss function for the training of the grasp prediction network has been developed and proved to work well also with low resolution 2-D images, then allowing the use of a single, smaller and low cost camera, that can be better integrated in robotic end-effectors. Despite the availability of less information (resolution and depth) a 75% of accuracy has been achieved on the Cornell data set and it is shown that our implementation of the loss function does not suffer of the common problems reported in literature. The system has been implemented using the ROS framework and tested on a Baxter collaborative robot.

**Keywords.** collaborative robotics, deep learning, vision-guided robotic grasping, engineering methods

## Introduction

Collaborative robots ("*co-bots*") are industrial robots able to safely operate within a workspace shared with human operators. Then, co-bots are conceived to aid and support human workers in uncertain environments, adapting to ever-changing scenarios, but always assuring safety. Co-bots may revolutionise industrial production, enablign the symbiotic collaboration of human workers and robots, but still lack of performance and their low financial return on investment limits their widespread application; the European Research Project Colrobot [1] aims at improving co-bots performance by developing a versatile mobile collaborative robotic platform, specifically conceived for automotive and aerospace assembly operations. To this purpouse, the ColRobot platform specifically addresses vision-guided grasping and dextereous manipulation of

[1] Corresponding Autor, Mail: margherita.peruzzini@unimore.it

many different types of objects in different environments. Artificial vision has proved to be collaborative robots' ideal sensing technology but its time consuming training drastically reduces the overall co-bots performance.

At state of the art, vision-guided grasping applications are based on *template matching* techniques, in which the images taken by vision sensors are processed, identifying the object and its grasping features in the scene, according to some reliable yet closed procedures such as *template Matching* (Figure 1). These methods rely on human intervention for the classification of every object of interest, and an hand-crafted definition of the correct grasp on every instance of the object. Moreover, the above procedure has to be repeated every time a new object is added, and human calibration of the system is needed if a major change in the environment happens.

Thus, state of the art industrial solutions are unfeasible when a higher degree of flexibility is required, as in the case of symbiotic human robot collaboration.

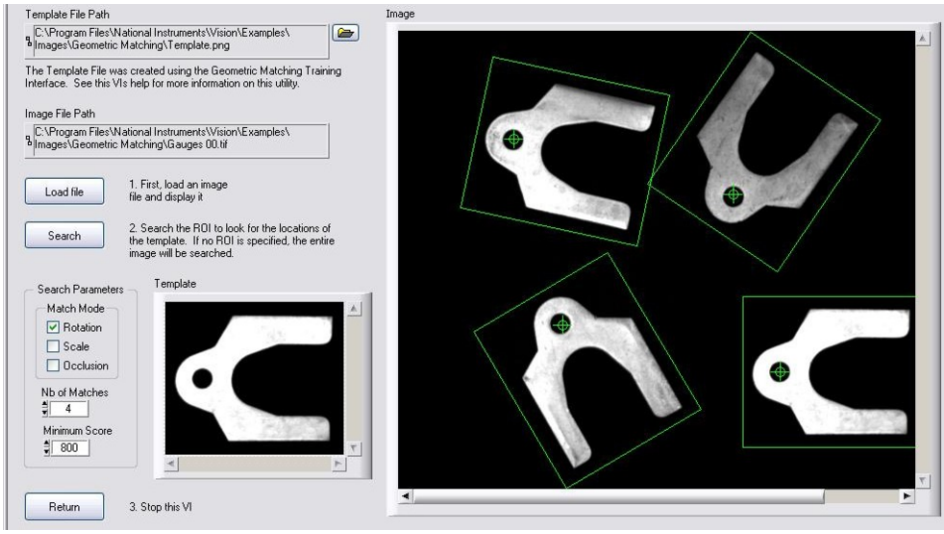


**Figure 1.** Industrial software example for Template Matching.

In recent years, Artificial Intelligence (AI) and deep learning (DL) has delivered massive improvements also in the field of computer vision [2]–[4], also enabled by the constant drops of the hardware costs. Therefore, there is a great opportunity to exploit the potential of deep learning techniques for the detection of the robotic grasps of unknown objects in an unstructured environment, which should enable collaborative robots to autonomously generate grasping strategies without the need of training and programming.

## 1. Related works

State of the art industrial robotics vision systems are based on standard computer vision techniques, in which the analysis is split into four main steps: pre-processing, feature extraction, reasoning/classification, reaction. In particular, the number and the types of the adopted features are manually selected for each specific application. For example, in Gleason et al. [5], the developed SRI vision module extract "blobs" from binary

images through a connected component analysis and then compute various moments on the object, in order to determine position, size and orientation. Such simple approach has the advantage to be fast and with high accuracy, but fails in case of overlapping objects and perform poorly with noisy images. In his patent, Roth [6] proposed a vision system that uses boundary features, such as lines, edges and holes, to recognize objects, thus overcoming the limits of the blob analysis with overlapping parts and low image quality. William [7] used Pattern Matching algorithm to produce a geometric description, i.e. real-valued position and orientation, of the detected feature. The descriptors were then used for Affine Searching procedure, in which 6 degree-of-freedom affine transformation on the geometry were applied, in order to detect variation in position, orientation, aspect ratio and skewing of the object of interest. Sanz et al. [8] implemented a system that uses a first step of global thresholding of the image for fast segmentation, then compute the principal moments of the object (centroid, orientation and inertia axis) from the boundary points. Finally, candidate grasp points are extracted and the most stable is chosen, based on off-line stability properties.

Concerning the Machine Learning approach to the grasping problem, recent works have put a lot of effort in dealing with the complexity of such task in a real world, unconstrained scenario, and most of them uses Supervised solutions like *Deep Artificial Neural Networks*.

An Artificial Neural Network (ANN) is a computational structure, made by several interconnected layers of single units called *Neurons*, that has the ability to learn different tasks without any previous knowledge or task-specific programming. During the learning process, the network is fed with examples that are manually labeled with the information that has to be learned, and the weights of the system, are changed in each iteration to better approximate the expected output, thus to progressively improve the performances on the task. Differently from traditional systems, the feature selection is done during the learning stage. The weights are modified to optimize a *Loss* function, that takes into account the prediction error of the network at the end of each iteration [9]. If the training set is sufficiently representative, after the training phase the network is able to generalize and perform well on the same task with new, unseen data.

A special architecture of an ANN is the Convolutional Neural Network (CNN), that is particularly suited for working on images or video streams as input. In a CNN the single unit is a filter, that moves on the image and is able to capture visual feature, such as vertical and horizontal lines, color gradients, curves and so on. Stacking such structure in layers allows the combination of those filter, and the progressively learning of more complex and thus semantically higher features (e.g., eyes, faces, street signs and so forth, depending on the task to be learned).

In order to better mimic the human grasping capabilities and to exploit visual sensors features, choosing a CNN architecture for the implementation of the task is straightforward. In Mahler et al. [10] a Deep CNN was trained on a synthetic dataset made of 6.7 million point clouds on several thousand of 3D models of objects, for the prediction of grasp poses, along with a robust analytic grasp metric that measure the probability of success of the grasping from the RGB-D images. The main drawback of this method is in the use of synthetic data to provide a wide enough set to train a Deep CNN. In fact, this choice could potentially lead to a loss of generalization on real-world images (i.e., the model may not be able to provide a correct grasp on real scenarios).

Lenz et al. [11] worked with the Cornell Dataset [12] (see Subect. 2.1), with two deep networks to evaluate all the candidate grasps. The first, faster network has fewer

features and is able to prune unlikely grasps, whilst the second is bigger and slower, but only needs to consider the top predictions of the first network.

Redmon et al. [13] proposed a Deep CNN architecture, derived from a widely adopted model from Krizhevsky et al. [14], named AlexNet, used for object recognition task. Their implementation uses pretrained AlexNet weights in order to speed-up the learning process, training the network on the above-mentioned Cornell Dataset thereafter. Such network was able to predict a subset of valid poses on the object, along with a classification of the object, added as a task to force the network to recognize similar objects, and thus predict similar grasping poses. Although these methods significantly improve the state of the art results on real data, they both lack the ability to adapt to unseen situations. In particular, given enough semantic noise in the scene (e.g., background light changes or partial objects in the scene), both methods tend to diverge from a valid grasp. This is related with the loss function used in both methods (i.e. the mean squared of the error between the ground truth and the prediction). For two areas of the image with strong gradients, the overall grasping prediction results in the average of two valid grasping poses, which is not guaranteed to be valid.

## 2. Grasping with Deep Neural Networks

In the next sessions we present our approach to the grasping problem in more details.

### *2.1. Dataset*

The Cornell Dataset [12], used aslo in the two main related works [11], [13], has been chosen as score baseline to compare the results of the work. The dataset provides 885 RGB images of 280 different objects, with several images taken of each object in different orientations or poses, along with an associated point cloud. Each image contains a single object, and a set of labels is provided, where each label is a rectangle depicting either a valid or an invalid grasp position on the object. Examples of the images on the dataset is shown in Figure 2.

**Figure 2.** Cornell Dataset example images.

The labeled grasping poses are defined for a two fingers industrial robotic gripper, and only a subset of all the viable poses on the object is provided. In the present work, we choose not to use the information coming from the point cloud data, as to push the development of a low cost solution, the uses only 2D low-resolution camera that can be better integrated in robotic end-effectors. To guarantee the robustness of the Deep Learning algorithm, as well as to generalize the knowledge built from the examples, several steps can be followed to augment the number of images that are fed to the network, in particular, starting from the raw dataset, the following modification were made to the images:

- the RGB images have been cropped to a fixed size window, centered around the object; the resulted image is a 224 x 224;
- both the valid and invalid grasping rectangles were translated according to the new, cropped window;
- each cropped image was randomly scaled *w.r.t.* the original size, choosing a scaling factor from a Gaussian Distribution with mean m=1 and variance $\sigma^2 = 0.15$;
- random translation and rotation were applied in a fixed range;
- changes in lighting using the YCbCr.

Finally, Gaussian noise was added to the images, to further enhance the robustness to sensor noise of the system. In Figure 3 a comparison between the original image and a subset of the processed images is shown.

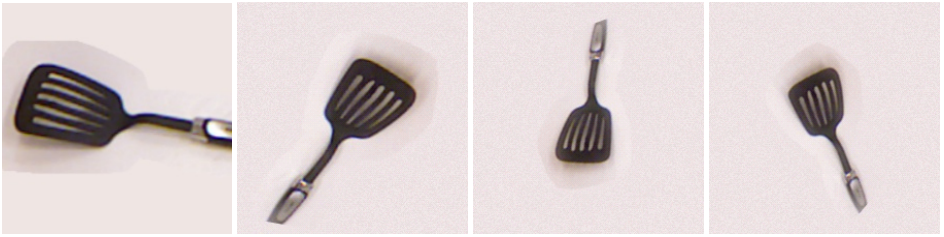

**Figure 3.** (Left) Image of the original dataset. (Right) Random scaling and translation of the original image.

Each label in the Cornell Dataset is a four-value tuple, containing the *(x,y)* value for each corner of the grasping rectangle. In order to assure the predicted output to be a rectangle, we transformed the coordinates in to a parametric representation, consisting of $\{x_c, y_c, w, h, \alpha\}$, where $(x_c, y_c)$ is the rectangle center coordinates, *w, h* are width and height and $\alpha$ the rotation of the main dimension of the rectangle *w.r.t.* the horizontal direction. Figure 4 better represents this parameters conversion.

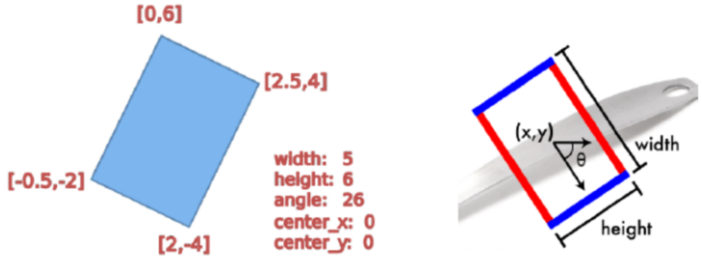


**Figure 4.** Parameters of a grasping rectangle.

## 3. Deep Neural Network implementation

### *3.1. Architecture*

Since the data avaialble were not enough to train a deep neural network in and end-to-end fashion, a model pre-trained on a different task with a wider dataset has been used. As listed on the main page, a subset of objects is common between our domain and the images contained in the ImageNet dataset [15], thus allowing us to start from the famous VGG-19 [16] architecture pre-trained for the image classification task. As such, the first layers have been frozen (i.e. we left their weights unchanged), as to exclude them from the training phase, while the last two were fully trained, as they contain high-level task-dependent information. Then, the architecture has been split in two branches; while the first one predicts the angle of the grasping rectangle, the latter predicts the remaining parameters (i.e. width, height and center coordinates). This is mainly due to the difference between the angle α and the other parameters, as the first ranges in [0°, 90°], while the others can take values up to the dimension of the image. The two branches are then joined, and a regression layer computes the grasping box corners coordinates from the five predicted parameters. The whole architecture is schematically shown in Figure 5.

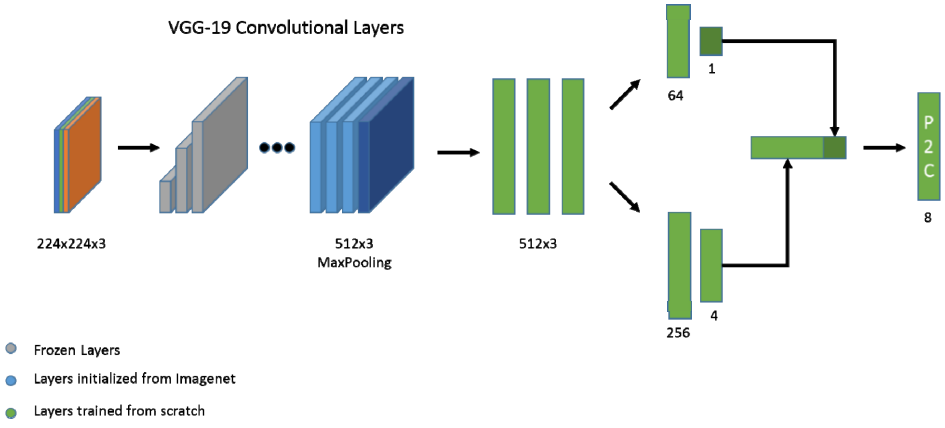


**Figure 5.** Proposed CNN architecture.

### *3.2. Loss function*

The literature presented in Section 1, related to deep neural networks for object grasping, shares the use of the MSE loss as main objectives during the training process. Such choice may not be ideal due to the following reasons:

- The learning objective differs from the accuracy score method, which is based on the Intersection Over Union metric;
- the MSE loss suffers from average results (i.e. for two different labels, the average of them is considered a good label). This hypothesis does not hold in every situation.

Thus, a new loss function based on the IoU metric is proposed, which is fully differentiable and it is shown to be consistent with the objective task of grasping. As shown in Figure 6, the loss is computed between two grasping rectangles, parameterized using 2D coordinates of the corners, and has values lying between 1 (perfect match between the two rectangles) and 0 (fully disjointed rectangles). A pseudo-algorithm of the loss computation is shown in Algorithm 1.

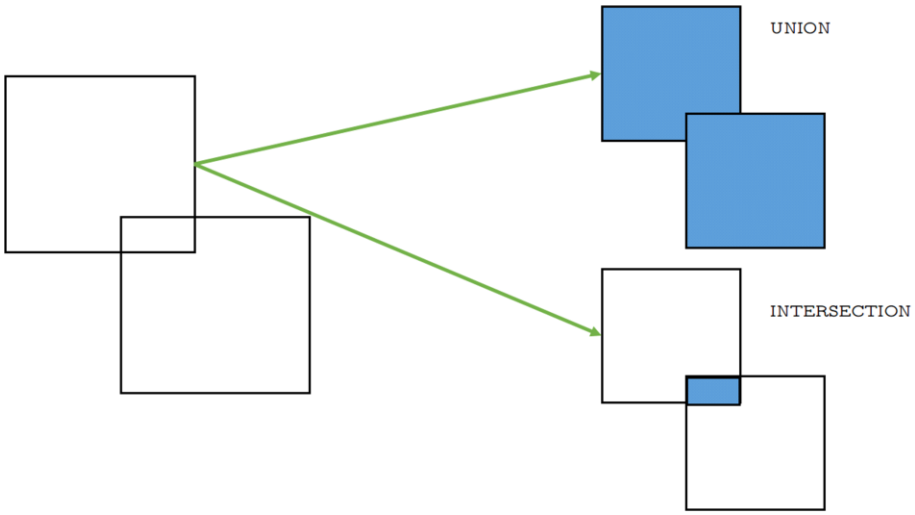


**Figure 6.** Geometric meaning of the IoU Loss function.

**Algorithm 1:** IoU Algorithm

**Data:** ground truth rectangle $r_t$ predicted rectangle $r_p$
**Result:** ioU score $\in [0, 1]$
$IoU = 1$;
**for** *both axes* $(X, Y)$ **do**
  project the 4 corner of $r_t$ and $r_p$ onto the axis;
  **if** *the minimum value belong to* $r_t$ **then**
    delete values from $r_t$ lesser than the minimum of $r_p$ ;
  **else**
    delete values from $r_p$ lesser than the minimum of $r_t$;
  **if** $r_t$ *projected is empty or* $r_p$ *projected is empty* **then**
    $IoU = 0$;
    end ;
  **if** *the maximum value belong to* $r_t$ **then**
    delete values from $r_t$ bigger than the maximum of $r_p$ ;
  **else**
    delete values from $r_p$ bigger than the maximum of $r_t$;
$Iou* = (value_{higher} - value_{lower})$

The implementation deeply relies on the Separate Axis Theorem (SAT), which scores the overlapping area of two convex polygons. SAT states that:

**SAT.** *If two convex objects are not penetrating, there exists an axis for which the projection of the objects will not overlap".*

In addition, SAT can also be used to compute a score of the IoU, accumulating the score of the length of the intersection segments along the two projected axis. For each training sample, the loss has been back-propagated between the predicted and the best overlapped pose. Alternative solution (e.g. to pick a random pose on every iteration) were also investigated, but the proposed method proved to deliver the best scoring results.

## 4. Results

### *4.1. Implementation Details*

The proposed implementation has been developed using the Tensorflow Framework [17] under Ubuntu Linux 14.04. The weights of the first layers have then been initialized from the ones of the VGG-19 network public available with the framework. The training batch size has been set to 128, training the network for 250 epochs with random data augmentation. From preliminary results, the batch size showed to be important for the convergence of the training phase, as it becomes unstable, due to the approximation of the current batch for the entire dataset: as it becomes larger, the network learns a better generalization. Adam [18] with learning rate starting from 0.0001 and exponential decay has been selected as optimizer. The network is saved every 5 epochs and only if the loss score improves. As for the data, it has not been emplyed the split method proposed in [13] that uses five cross validation, preferring to simply split the dataset using a 80:20 proportion image wise.

### *4.2. Simulation Results*

The accuracy of the method has been tested using the mean IoU directly, in contrast with the literature [11], [13]. A proposed grasping pose has been considered valid if the IoU with any of the annotated poses for the image scored more than 50%, since this method fits better the nature of the proposed loss function. Results are reported in Table 1.

**Table 1.** Score for IoU and Accuracy measures on both train a test.

| Metric | Train (%) | Test (%) |
|---|---|---|
| Mean IoU | 0.775 | 0.625 |
| Mean Accuracy | 0.85 | 0.734 |

### *4.3. Experimental Results*

In this subsection a possible end-to-end implementation for a collaborative robotic grasping is presented.

In the developed application, the robot is deployed on a shop floor, facing a table filled with hand tools and fixtures. No constraints are given on the number of objects on the table or on the lighting condition of the environment.

An operator is performing a task together with the robot. The pipeline of the application is the following:

1. The operator asks the robot for an hand tool or a part for the task he is performing;
2. the vocal request of the operator is translated into text, by means of a off-the-shelf Speech Recognition software, such as Google Voice-To-Text Service;
3. the application launch an Object Recognition and Localization procedure, based on state-of-art Deep CNN for localization, such as Y.O.L.O [19];
4. the requested item is found among the visible ones in the scene seen by the robot, and an image cropped around the localization coordinates of the object is sent to our Grasping Network;
5. the Grasping Network predicts the grasping rectangle coordinates;
6. the robot moves accordingly to reach and safely grasp the object.

A video demonstration of the application can be seen at [20].

## 5. Conclusions and future development

In this work, a Deep Learning approach to the industrial problem of robotic grasping is presented. A novel CNN architecture is trained on a small dataset, reaching about 73% score using less data than the state-of-art related works, encouraging further research and improvement on the system.

The original contributions of the method are twofold:

1. a novel definition of a Loss Function, which does not suffer of common problems found in related literature;
2. the algorithm only uses information coming from a low-cost, 2D camera, making the integration of the system easier on a industrial environment, such as a robotic cell or collaborative platforms.

Future developments will focus on integrating the object recognition and localization pipeline with the grasping prediction in one single architecture, with the purpouse to reduce the *context switching* overhead in the implementation of the end-to-end application, and the extension of the grasping prediction in cluttered environment.

## Acknowledgement

This paper is supported by European Union's Horizon 2020 research and innovation program under grant agreement No. 688807, project ColRobot (Collaborative Robotics for Assembly and Kitting in Smart manufacturing)

## References

[1] Colrobot Consortium, *Colrobot*, 2016. www.colrobot.eu. Accessed Feb, 10 2018.

[2] C. Dong, C.C. Loy, K. He and X. Tang, Learning a Deep Convolutional Network for Image Super-Resolution, in *Computer Vision–ECCV 2014*, Vol. 8689, 2014, pp. 184–199.

[3] G. Bertasius, J. Shi, and L. Torresani, DeepEdge: A multi-scale bifurcated deep

network for top-down contour detection, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, Vol. 7, pp. 4380–4389.

[4] O. Vinyals, A. Toshev, S. Bengio and D. Erhan, Show and tell: A neural image caption generator, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, Vol. 7, pp. 3156–3164.

[5] G.J. Gleason and G.J. Agin, A modular vision system for sensor-controlled manipulation and inspection, in *Proceedings of the 9th International Symposium on Industrial Robots, Washington D.C., USA*, 1979, pp. 57–70.

[6] S. D. Roth, Vision system for distinguishing touching parts, Patent #4,876,728, 1989.

[7] W. Silver, Geometric Pattern Matching for Industrial Robot Guidance, in *Robotics Research*, 2000, pp. 69–77.

[8] P.J. Sanz, A. Requena, J.M.I. Quereda, and A.P. Del Pobil, Grasping the not-so-obvious: vision-based object handling for industrial applications, *IEEE Robot. Autom. Mag.*, Vol. 12, 2005, No. 3, pp. 44–52.

[9] G. Cybenko, Approximation by superpositions of a sigmoidal function, *Math. Control. Signals, Syst.*, Vol. 2, 1989, No. 4, pp. 303–314.

[10] J. Mahler *et al.*, Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics, *arXiv Prepr.*, 2017.

[11] I. Lenz, H. Lee, and A. Saxena, Deep learning for detecting robotic grasps, *Int. J. Rob. Res.*, Vol. 34, 2015, No. 4–5, pp. 705–724.

[12] I. Lenz, H. Lee and A. Saxena, Cornell Grasping Dataset, 2013, http://pr.cs.cornell.edu/grasping/rect_data/data.php, Accessed: Jan, 31 2018.

[13] J. Redmon and A. Angelova, Real-Time Grasp Detection Using Convolutional Neural Networks, *Robot. Autom. (ICRA), 2015 IEEE Int. Conf.*, 2015, pp. 1316–1322.

[14] A. Krizhevsky, Ii. Sulskever and G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, *Adv. Neural Inf. Process. Syst.*, Vol. 60, 2012, No. 6, pp. 84–90.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2009, DOI: 10.1109/CVPR.2009.5206848.

[16] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, *arXiv Prepr. arXiv1409.1556*, 2014.

[17] Martìn Abadi *et al.*, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, White paper, 2015, Accessed: Jan, 31 2018, https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/45166.pdf.

[18] D. P. Kingma and J. Ba, Adam: A Method for Stochastic Optimization, *ICLR 2015*, https://arxiv.org/abs/1412.6980v5.

[19] J. Redmon and A. Farhadi, YOLO9000: Better, Faster, Stronger, *arXiv Prepr. arXiv1612.08242*, 2016.

[20] Xilab, Colrobot Te2018, 2018. https://www.youtube.com/watch?v=6DpXqTWLNwo, May, 4 2018.