ISSN 1213-6972 Journal of WSCG http://www.wscg.eu Vol.28, No.1-2, 2020

Mixing deep learning with classical vision for object recognition

Maciej Stefańczyk
Warsaw University of Technology
Institute of Control and Computation Eng.
Nowowiejska 15/19, 00-665 Warsaw, Poland
maciej.stefanczyk@pw.edu.pl

Tomasz Bocheński
Warsaw University of Technology
Institute of Control and Computation Eng.
Nowowiejska 15/19, 00-665 Warsaw, Poland
tbochens@gmail.com

ABSTRACT

Nowadays, when one needs a system for image recognition, it is mostly a matter of finding pre-trained CNN and, sometimes, adding additional training based on transferred knowledge. Accurate 6-DOF object localization in the image is a more laborious task and requires more complex training data to be available. On the other hand, if we know the model of the object, it is straightforward to acquire its pose from the image (RGB or RGB-D). In this paper, we try to show the advantages of mixing deep learning object recognition/detection with classical 6-DOF pose estimation algorithms, with a focus on applications in service robotics.

Keywords

CNN object detection, VGG16, ResNet50, 6-DOF pose estimation, RanSaC, ICP, RGB-D

1 INTRODUCTION

1.1 Motivation

It is hard to imagine life without robots. They have become an inseparable part of our reality and are applicable in almost every area of life. Robots are used in factories for transport, production, and quality control. Telemanipulators, controlled by doctors, are used to perform surgical procedures. Thanks to robots, hazardous environments (for example, underwater or in space) can be safely investigated. In the military, their primary use is to disarm bombs or take other dangerous actions, saving the life of the soldiers.

Robotics, as an industrial field, is developing rapidly. According to a report of the International Federation of Robotics, global robot sales in 2017 increased by 30% compared to the previous year [oR18]. The development in the field of industry is also accompanied by great progress in the field of research. Modern robots are able to carry out work that not long ago was only performed by people. One of the sources of progress is equipment robots with senses, making them more autonomous.

Eyesight is one of the most important senses of man as it allows us to perceive most of the information from the environment. It is estimated that 83% of human perception takes place through sight. For comparison, hearing, smell, touch, and taste are processing 11%, 3.5%, 1.5%, and 1% of information respectively [SK11]. Therefore, equipping robots with eyesight becomes an important research issue.

One of the tasks in service robotics is object manipulation, where robots should cope with everyday things from the human surrounding. For this task to be performed flawlessly, robots must be equipped with effective manipulators and grippers and a vision system capable of accurate object pose estimation. Deep learning approaches have proven their high quality and accuracy in object classification multiple times, and are capable of distinguishing hundreds or thousands of different classes with minimal impact on performance. Accurate object pose estimation, on the other hand, is straightforward to achieve if we know what object we see and we have its model. In this paper, we try to show the advantages of mixing very popular deep learning approaches for object classification/detection with classical 6-DOF pose estimation algorithms in service robotics applications.

1.2 Paper structure

The rest of the paper is structured as follows. The next section describes state of the art in object detection and pose estimation tasks. Section 3 presents proposed system structure, followed by the implementation details in sec. 4. Sample results for a single scenario are presented in sec. 5, and the last section concludes the paper.

2 STATE OF THE ART

2.1 Object classification and detection

Object classification answers the question *what is on this picture*. Object detection deals with a more complicated task of not only telling *what* is on the picture, but also *where*, for multiple objects at the same time. The simplest way of creating an object detector is to apply the classifier multiple times (e.g. sliding window) on different input scales, but there are also much more sophisticated techniques. Here the important algorithms for both are presented. The advent of object detection can be connected with the first Viola-Jones face detector [VJ01], which, although being still the sliding window detector, applied methods for quick rejection of candidate regions to achieve big performance gain. Next step in "classical" object detection is HOG [DT05] with further improvement of deformable part models [FMR08]. Those solutions were very good at their time, and to some extent, were able to generalize on objects different from the initial ones (faces and humans). As they rely mostly on shape, not texture, it is hard to use them to distinguish between different subtypes of similar objects.

Another branch of classification and detection algorithms is based on the image feature points (either classical SIFT [Low99] or more recent binary descriptors [RRKB11]). Those can be either compared directly with the object models (with RanSaC) or their statistics can be used (in the form of a bag of visual words). The latter can be further extended to create an object detector [VL09].

All aforementioned approaches were model-based, handcrafted to a smaller or bigger extent. With the advent of consumer-grade high power GPUs, the branch of data-driven approaches emerged, with neural networks and deep learning playing the main role. In classification task, the most widely used architectures are VGG [SZ14], Inception [SVI$^+$16] and ResNet [HZRS16], with their extensions and new ideas being published almost constantly [ZSGY19].

Currently, two types of detectors are used to solve the problem of object detection with the use of CNN, i.e. one-stage and two-stage. The way of operation of two-stage detectors consists of two steps: proposing the regions of interest and classification of these, together with the improvement of their bounding boxes. The single-stage detectors immediately determine the bounding boxes along with the classes of objects, without first detecting the regions of interest. The two-stage detectors are characterized by high performance in terms of accuracy but do not always operate in real time. The single-stage detectors operate in real-time, but the results are not always as good as those of the two-stage detectors. The most popular two-stage detector is Faster R-CNN [RHGS15], while the most commonly used single-stage detectors are SSD [LAE$^+$16] and YOLO [RDGF16].

2.2 Pose estimation

The goal of object pose estimation is to find the position and orientation of the query object in the scene (relative to some given coordinate frame, e.g. camera). This task can be performed either solely in RGB space or using additional depth information provided by current sensors (like Kinect or Intel RealSense). Contrary to object detection (or classification), in general, one needs an exact model of the object to find its pose.

A simple (yet effective) approach to pose estimation for textured objects relies on feature points. Those can be either extracted directly from RGB (like SIFT or ORB) and then matched against a set of reference views [Low01] or reprojected to 3D using available depth information and matched against reference sparse cloud of feature points. When depth (or 3D in general) information is available, features can be computed directly from 3D [MBO10, RBTH10]. To overcome problems from multiple matches between the feature points (in multi-object scenarios), additional hypothesis verification can be applied [HCL$^+$13, ATP$^+$13].

For texture-less models, LineMOD [HLI$^+$12] uses multiple object silhouettes, generated from the rendered 3D views. Part-based models, along with 3D CAD models, can also be efficiently used to retrieve the pose of challenging objects [AME$^+$14]. An interesting branch is also usage of procedural models [GFJ$^+$18], which, apart from the pose, estimates also other object parameters, like size, radius, etc.

The next advancement in the field was the application of trainable RGB-D feature detectors. It is either based on previous solutions (like extending LineMOD [TTKK14]) or learning features with the CNN approaches [GAGM15, KMT$^+$16].

Finally, end-to-end deep learning solutions were proposed. This problem can be simplified if only the grasp region candidates are required [TTS$^+$18]. PoseCNN [XSNF17] extracts full 6D pose of the 21 known objects. The authors subdivided the problem into three main steps: semantic labeling, 3D position estimation, and final rotation regression, implemented as the set of components in a multi-task network. The network was trained on the data generated from the YCB dataset [CSW$^+$15], providing textured 3D models of the objects and perfectly labeled test scenes. SSD-6D [KMT$^+$17] is an approach to extend the SSD network to work with 3D data. In order to facilitate the final 6D pose estimation, the authors trained the network to recognize one of the multiple discrete viewpoints. Final transformation refinement is done using classical vision algorithms (based on either RGB-only or depth-enriched data). One of the latest

advancements in the field is YOLOff [GKMM20], which is also a pair of cooperating networks. First detects the 3D object patches, which are then passed through the 3D points regression module, detecting the corresponding points between the scene and the model. Final pose refinement is done using classical algorithms. Those kinds of solutions are, however, much harder to train than 2D object detectors and not as popular yet [HMB+18]. As the authors of [XSNF17] state, sufficient and well-labeled training data is necessary. On the other hand, for classical approaches, only a simple model is enough, and no retraining is needed in case new models are required.

3 PROPOSED SYSTEM STRUCTURE

To take the best of two worlds, we propose the hybrid system, with CNN based object detector and feature-point based pose estimator (as a lot of the objects in human surroundings possess a rich texture). To further increase the system performance, we decided to use additional depth data (from the RGB-D sensor mounted on the robot). The system is composed of three main processing blocks (fig. 1) and an additional 3D model database.

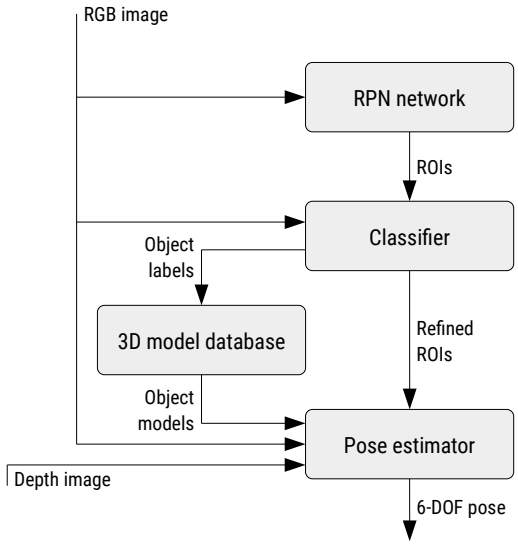


Figure 1: System structure

The first block is responsible for regions of interest detection in the RGB image. Service robots usually cope with a small number of objects visible in a single scene, and typically don't require high detection framerates. In that kind of scenario, separating RPN from classifier doesn't impact the overall system performance. Detected regions are passed to the classifier, which labels the ROIs with known classes and refines the bounding boxes. Finally, the object's pose is estimated using additional depth data and the 3D model retrieved from the database based on the label provided by the classifier.

4 IMPLEMENTATION DETAILS

4.1 Region proposal network

The region proposal network was based on simple yet popular VGG16 architecture. This architecture provides a good compromise between complexity (has only 16 trainable layers) and quality (won the ILSVRC competition in 2014). Simplified scheme of our RPN is presented in fig. 2.

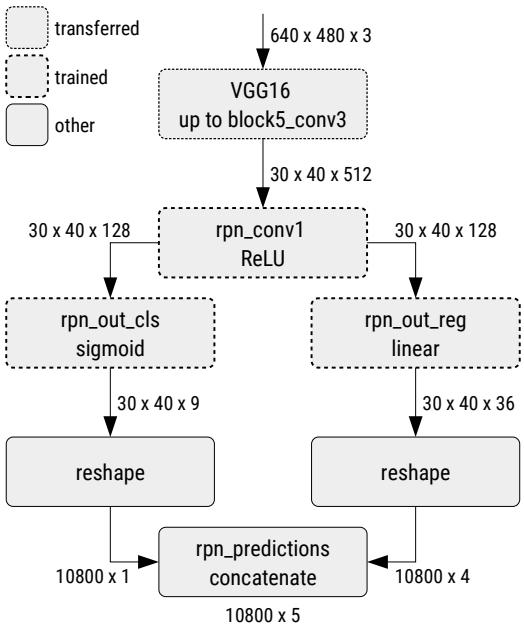


Figure 2: Region proposal network

The network takes VGA RGB images as the input. There were nine anchors defined for RPN. As the feature extracting backbone, the first 13 layers from VGG were utilized, with an additional convolution added. After that, the network splits into classification and regression parts. Classification part produces a label (object/background) for each anchor box (9 in total). The regression part produces offsets (for top left corner) and size corrections (width and height) for each anchor box.

4.2 Classification network

For the classification stage, the ResNet50 was used as a backbone (fig. 3). Object proposals, cropped to 224×224 pixels, were fed as an input. Two additional convolution layers further transformed features from the backbone, and the global average pooling layer produced final features. After that, network splits into the classification part (fully connected, all known classes plus one for background) and bounding box regression part (also fully connected, four values, like in RPN).

4.3 Pose estimation

After the object detection step, all recognized objects are cropped from the RGB image along with accompanying depth information. Object type information

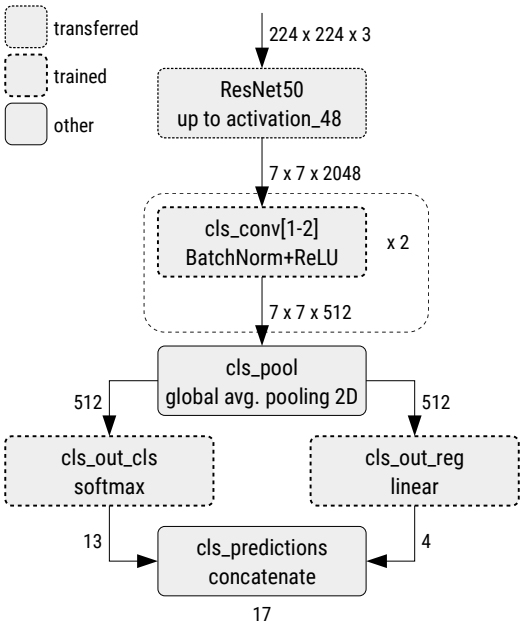


Figure 3: Classification and localization network

is used to retrieve the appropriate 3D model from the model database (both dense point cloud and sparse feature cloud). Pose estimation is divided into three main steps (fig. 4). The first one is responsible for feature extraction and matching. On the cropped RGB image, SIFT feature points are detected, which are then reprojected from 2D image coordinates to 3D world coordinates using available depth information. After that, those are matched with model features.

When the pairs of matched points are prepared, the RanSaC is used to estimate the initial rigid transform between the model and the query image. As this transform is based on a sparse cloud of features, and the query cloud itself is very low resolution, this transform often needs refinement. To do this, as the last step, ICP is used. It tries to find the final transformation between the full model point cloud and the query point cloud (built from RGB and depth crops). In the testing scenario, simple ICP with the point-to-plane metric was used.

5 RESULTS

5.1 Scenario, hardware and dataset

The prepared system was tested on everyday objects that can be found in the kitchen, mainly tea boxes and food cans. The assumed scenario was a robot tasked to find and bring a particular object from the cupboard. As the system should detect the same kind of objects that the robot works with, it was trained on some most popular ones. The set consists of 12 different objects, and, as there is no available dataset with those, the new one had to be created. For the training phase of the detector, there were around 30 images for each object,

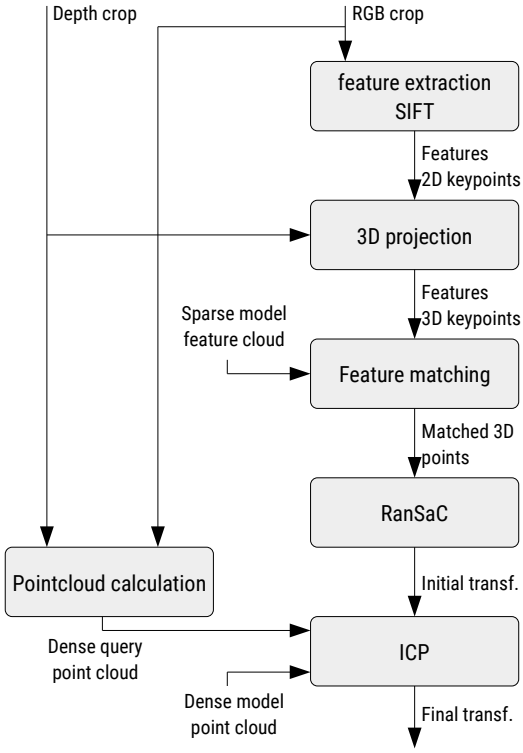


Figure 4: Pose estimation module

taken from two angles (front and top) around the object. For each image, there was also a mask prepared for further cropping and preparing augmented training dataset. Data acquisition setup and sample object cutout are presented in fig. 5. Pose estimation step used 3D object models prepared as cuboids or cylinders, with applied scanned textures (1 model per object). The whole process of data acquisition follows the one described in [KS17], and created datasets are publicly available [SLK16].

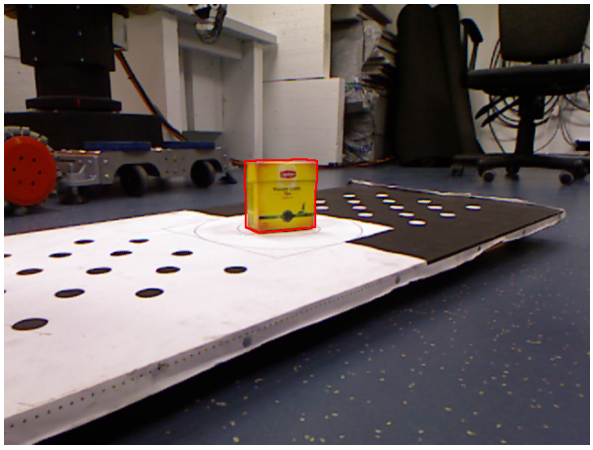


Figure 5: Training data acquisition – sample image with generated mask overlay

The data was acquired using the MS Kinect sensor, providing aligned RGB and depth images with VGA res-

olution. Test scenes were arranged with the objects placed inside the cupboard (fig. 6).

5.2 Detection

On the object detection stage, two parts of the network were evaluated. RPN network was trained using weights transferred from the ImageNet model, and only the added layers' weights were optimized using Adam optimizer with a learning rate of $10e^{-4}$. 10000 synthetic images were used for training. The final network achieved a precision of 0.63 and a recall of 0.98. High recall suggests that all interesting objects were properly selected. Low precision means that apart from the objects of interest (12 classes), others were selected (yellow boxes on fig. 6). This is a good result, as it makes the network robust to changing needs and easy to extend with new objects. The spurious RPN detections are filtered on the classification stage. On 4GB GTX 970 (rather mediocre in today's standards), training took 10 hours. This time is not very short, but the RPN doesn't have to be retrained if new objects are added to the system.

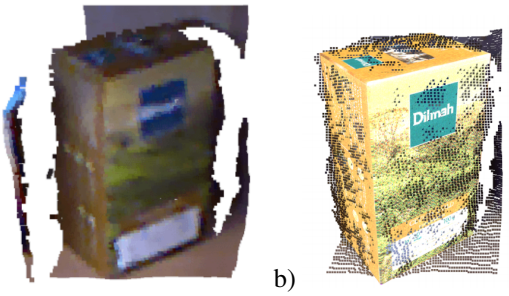
Figure 6: Sample result of object detection: yellow boxes – RPN result, green boxes – detected objects.

The classifier also had its initial weights transferred from the ImageNet model, and also only the top layers had updated weights. The training was done using 2000 synthetic images (taken from the same set as in RPN training), and the network achieved its final quality after only three epochs. This fact makes the process of adding new objects very fast. Classifier network achieved 0.99 precision and 0.98 recall. The training took 3.5 minutes on GTX 970.

Final tests were conducted for the network cascade, where the output of the RPN was tied to the classifier input. This time the precision was still very high (0.99), but recall dropped to 0.92 (mainly due to one class, which had the worst statistics in both the RPN and classifier stage). Sample detections are marked with green boxes on fig. 6. The top-middle object is not in the training set, so it was properly ignored by the classifier.

5.3 Pose estimation

The output of the detector is used to cut the single objects from the RGB and depth image. Due to the limited resolution of the Kinect sensor, the resulting object clouds contain only a small number of points (fig. 7a). It is, however, still possible to fit the 3D model to this cloud. For the fitting stage, the initial estimation is done using the RanSaC transformation estimation on the sparse cloud of SIFT points. Features are calculated on the RGB image, and their 3D coordinates are calculated based on the depth image. In 3D models, a sparse cloud of SIFT points is created on the model creation stage in a similar way [KL16].



a) b)

Figure 7: 6-DOF pose estimation: a – input, b – result

After the initial transformation is known, the full point clouds of the object cropped from the scene and the model are used in ICP refinement. Final transformation (fig. 7b), along with the known object parameters (e.g. its size) is then passed to robots grasping subsystem [SS16].

6 CONCLUSION

6.1 Results discussion

The presented system combines the advantages of commonly used detection and pose estimation algorithms. RPN network is trained in such a way, that it is possible to detect novel objects from similar classes (new boxes or cans), without the need for further training. Classification module, in order to correctly recognize particular instances, has to be trained on data as close to the target as possible, but as only a few layers are trained there, this process is rather fast. This makes the process of extending/modifying known objects dataset doable in an acceptable time (3.5 minutes in our experiments). Pose estimation part works with the already preselected data – object that is on the scene is already known and only the 6-DOF pose has to be fit. Adding new objects requires no retraining of this stage, as it is purely procedural.

Performance-wise, the object detection part takes 0.9 s per image and pose estimation another 1.0 s (on average, depending on the spatial resolution of the object). The whole process is definitely not real-time, but for

the grasping purpose, it is more than enough. After the first phase (detection) coarse pose of the object can be calculated, and while the robot moves its arm to the pre-grasp location, the final object pose is calculated. The system was tested on the Velma service robot [WBS15], working in a safe environment (fig. 8).

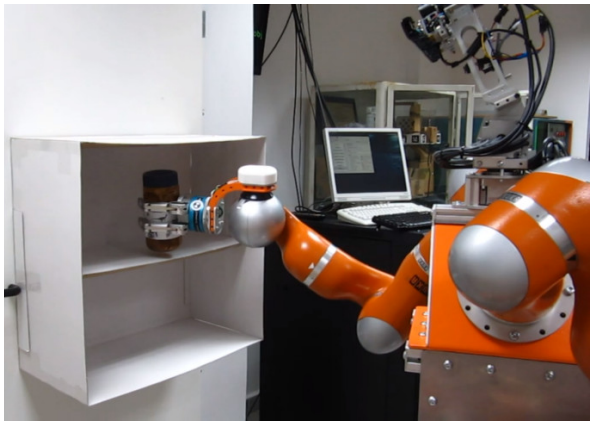


Figure 8: Robot grasping the object

It is hard to directly compare our solution with those presented in sec. 2. The most crucial difference, which makes the comparison unfeasible, is the format of the training dataset. PoseCNN, SSD-6D, and similar requires pose-labeled images as the input for the training. We decided that to make the retraining and extending the system easier, those are not required (and we don't have them). On the other hand, one of the strong assumptions in the presented solution is the rich texture in the objects. In datasets used by the aforementioned methods, those are a minority.

One comparison can be made in terms of the expected burden of system extension with new objects. Data-wise, in our system, one has to supply few (less than 20) pictures of the new object and scanned textures (as the many household objects can be modeled with simple shapes). In contrary, to train the end-to-end solutions, one has to supply pose-labeled pictures, and there must be a lot of them to cover most of the possible viewpoints. Time-wise, as it was described above, only the part of the pipeline needs to be retrained, with a smaller amount of data, which should take less time than retraining the full end-to-end pipelines.

6.2 Future works

There are multiple ways this system can evolve. For the training phase, instead of using the pictures of the object, one can use the rendered scenes solely [PZL+18], with perfect masks and no need for hand labeling.

Another possible direction for making the pose estimation better is changing the feature points detector to binary (if algorithm speed is crucial) or apply depth-based feature points rectification [Ste18]. ICP transformation can also be refined by using other ICP flavors (e.g. color information [LKS16]).

In its current form, the system is general-purpose and can be applied in multiple scenarios. As it is used in robotic applications, some extensions facilitating the hardware available can be made. The biggest impact on the pose estimation accuracy can be obtained using the active vision. After the first estimation, when the arm moves towards the object, additional pictures from the wrist-mounted sensor can be passed to the pose refinement subsystem, effectively implementing the multi-camera visual servoing [KZ15]. As the initial pose is already known, only a few ICP iterations could make the estimation better.

7 ACKNOWLEDGEMENTS

M. Stefańczyk is funded by the National Science Centre, Preludium grant no. UMO-2017/25/N/ST6/02358.

8 REFERENCES

[AME+14] M. Aubry, D. Maturana, A. A. Efros, B. C. Russell, and J. Sivic. Seeing 3D chairs: exemplar part-based 2D-3D alignment using a large dataset of cad models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3762–3769, 2014.

[ATP+13] A. Aldoma, F. Tombari, J. Prankl, A. Richtsfeld, L. Di Stefano, and M. Vincze. Multimodal cue integration through hypotheses verification for RGB-D object recognition and 6DOF pose estimation. In *2013 IEEE international conference on robotics and automation*, pages 2104–2111. IEEE, 2013.

[CSW+15] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar. The YCB object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015.

[DT05] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[FMR08] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[GAGM15] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Aligning 3D models to RGB-D images of cluttered scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4731–4740, 2015.

[GFJ+18] R. Getto, K. Fina, L. Jarms, A. Kuijper, and D. W. Fellner. 3D object classification and parameter estimation based on parametric procedural models. *WSCG*, 2018.

[GKMM20] M. Gonzalez, A. Kacete, A. Murienne, and E. Marchand. YOLOff: You Only Learn Offsets for robust 6DoF object pose estimation. *arXiv preprint arXiv:2002.00911*, 2020.

[HCL+13] Q. Hao, R. Cai, Z. Li, L. Zhang, Y. Pang, F. Wu, and Y. Rui. Efficient 2D-to-3D correspondence filtering for scalable 3D object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 899–906, 2013.

[HLI+12] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pages 548–562. Springer, 2012.

[HMB+18] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, et al. Bop: Benchmark for 6d object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 19–34, 2018.

[HZRS16] 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*, pages 770–778, 2016.

[KL16] T. Kornuta and M. Laszkowski. Perception subsystem for object recognition and pose estimation in RGB-D images. In *Recent Advances in Automation, Robotics and Measuring Techniques*, volume 440 of *Advances in Intelligent Systems and Computing*, pages 597–607. Springer, 2016.

[KMT+16] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab. Deep learning of local RGB-D patches for 3d object detection and 6d pose estimation. In *European conference on computer vision*, pages 205–220. Springer, 2016.

[KMT+17] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1521–1529, 2017.

[KS17] T. Kornuta and M. Stefańczyk. ModReg: a modular framework for RGB-D image acquisition and 3D object model registration. *Foundations of Computing and Decision Sciences*, 42(3):183–201, 2017.

[KZ15] T. Kornuta and C. Zieliński. Robot control system design exemplified by multi-camera visual servoing. *Journal of Intelligent & Robotic Systems*, 77(3–4):499–524, 2015.

[LAE+16] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[LKS16] M. Łępicka, T. Kornuta, and M. Stefańczyk. Utilization of colour in ICP-based point cloud registration. In *Proceedings of the 9th International Conference on Computer Recognition Systems CORES 2015*, Advances in Intelligent Systems and Computing, pages 821–830. Springer, 2016.

[Low99] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[Low01] D. G. Lowe. Local feature view clustering for 3D object recognition. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.

[MBO10] A. Mian, M. Bennamoun, and R. Owens. On the repeatability and quality of keypoints for local feature-based 3D object retrieval from cluttered scenes. *International Journal of Computer Vision*, 89(2-3):348–361, 2010.

[oR18] International Federation of Robotics. Executive summary world robotics 2018 industrial robots, 2018.

[PZL+18] J. Peng, C. Zheng, P. Lv, T. Cui, Y. Cheng, and S. Lingyu. Using images rendered by PBRT to train Faster R-CNN for UAV detection. *WSCG*, 2018.

[RBTH10] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3D recognition and pose using the viewpoint feature histogram. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2155–2162. IEEE, 2010.

[RDGF16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[RHGS15] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[RRKB11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011.

[SK11] H. D. Stolovitch and E. J. Keeps. *Telling ain't training*. American Society for Training and Development, 2011.

[SLK16] M. Stefańczyk, M. Laszkowski, and T. Kornuta. WUT visual perception dataset: a dataset for registration and recognition of objects. In *International Conference on Automation*, pages 635–645. Springer, 2016.

[SS16] D. Seredyński and W. Szynkiewicz. Fast Grasp Learning for Novel Objects. In *Recent Advances in Automation, Robotics and Measuring Techniques*, volume 440 of *Advances in Intelligent Systems and Computing*, pages 681–692. Springer, 2016.

[Ste18] M. Stefańczyk. Improving RGB descriptors using depth cues. In *Computer Vision and Graphics*, volume 11114 of *Lecture Notes in Computer Science*, pages 251–262. Springer, 2018.

[SVI+16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[SZ14] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[TTKK14] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim. Latent-class hough forests for 3D object detection and pose estimation. In *European Conference on Computer Vision*, pages 462–477. Springer, 2014.

[TTS+18] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. *arXiv preprint arXiv:1809.10790*, 2018.

[VJ01] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001.

[VL09] V. Viitaniemi and J. Laaksonen. Spatial extensions to bag of visual words. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, pages 1–8, 2009.

[WBS15] T. Winiarski, K. Banachowicz, and D. Seredyński. Two mode impedance control of Velma service robot redundant arm. In *Progress in Automation, Robotics and Measuring Techniques. Vol. 2 Robotics.*, volume 351 of *Advances in Intelligent Systems and Computing*, pages 319–328. Springer, 2015.

[XSNF17] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[ZSGY19] Z. Zou, Z. Shi, Y. Guo, and J. Ye. Object detection in 20 years: A survey. *arXiv preprint arXiv:1905.05055*, 2019.