

Robotic Vision with Deep Learning

Kursad UCAR
Selcuk University

Hasan Erdinc KOCER
Selcuk University

Introduction

A robot is a mechanical device that can automatically perform a complex sequence of actions and is usually programmable by a computer. Internal or external controllers can control the robots. Robots can be created to resemble humans or animals, but most robots are designed to perform a task regardless of their appearance (Ruiz-del-Solar et al., 2018). As the same time robots are controlled fully or semi-autonomously in terms of being controlled.

Robotics is the field of technology that deals with the design, manufacture, operation, and application of robots (Samuel E. Anthony, 2016). Robots are machines that can replace humans in hazardous environments or production processes. For this reason, they are also more similar to humans in terms of appearance, behavior, or cognition. In addition to humans, there are also robots inspired by nature, such as spiders and worm robots. With these robots, a new robotic field was born under the name of soft robotics. Robotics researchers are trying to use robots in many different tasks. Some tasks can be summarized as acting like a human, garbage sorting, automatic direction finding and navigation, collaborative automation, debris search and rescue, and drawing pictures. Deep learning will play an active role in all these tasks. The difficulties in this and similar tasks can be briefly summarized as follows.

- 1) Advanced manipulation. Grasping of an object may differ according to the object. It is very hard to grasp things with complex shapes, which can break or change shape when squeezed. Likewise, tasks required by the grasping process, such as painting, opening a door, or using a tool, are of interest to researchers.
- 2) Learning dynamics: It has always been a problem for robots, as changes force them to step outside of the algorithm that controls them. Monitoring and capturing changes requires human expertise and is time-consuming. It is significant to adapt to new dynamics in order to solve problems such as grasping objects that are different from the ones taught, and moving in a new environment.
- 3) Sensor fusion & dimensionality reduction. Robots need to be fast to be used in real-time applications. For this reason, obtaining meaningful data from low-quality images also plays a critical role. Thanks to deep learning, the data can be represented and used with data of lower dimensions by cleaning it from data that

will not be used.

- 4) Interpreting and anticipating human actions. Robots can also interact with humans. In emergency rescues, robots that are put into the service of people, such as elderly / sick care, should work in a way that does not harm people. In such tasks, robots must also be able to predict human movements.
- 5) Advanced object recognition. Robots deal with a wide variety of objects. Object recognition and classification are some of the things that deep learning does best. Thanks to deep learning, it has become possible to recognize objects and obtain their properties, to make necessary path calculations, to obtain critical properties (slippery, sharpness, etc.) and surface characteristics of objects.

A robot perceives the world it is in (simulation or real) with its different sensors, creates a model of the world and updates this model over time when necessary. However, in order for the robot to perform a task, it is essential to make decisions, plan the necessary actions and implement these actions. While performing these actions, it is highly adhered to the program written by the human. Keeping the robots working without being affected by environmental or parameter changes is still not an insurmountable problem. Machine learning is a useful tool to overcome the mentioned and similar challenges. In particular, deep learning has become well known for its outstanding success in many fields.

Robot Kinematics

A robot is structurally created with joints that perform translational (prismatic-P) and rotational (revolute-R) motions, as shown in Figure 1, and joints that connect these joints to each other. The displacement due to the rotational movement and the translation due to the displacement between the joints are called the joint angle and the joint offset, respectively. Robots are positioned by controlling the joints separately. Thus, the positioning of the endpoint is ensured. Movement in Cartesian coordinates is achieved by obtaining the joint coordinates that will provide the current position using kinematic equations.

Kinematics is a science of motion that deals with the motion of objects. The masses of the objects and the forces acting on the object are not taken into account in the movements discussed. In robot kinematics, a relationship is defined between the end-effector and the joints that provide the movements.

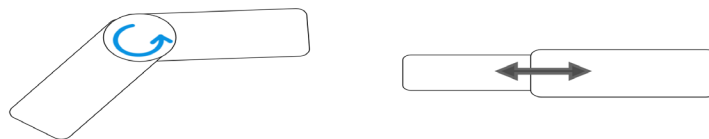


Figure 1. Prismatic and revolute joints

The position and orientation of each joint of a robot is determined relative to the previous or the next joint. This consecutive relationship is called an open kinematic chain.

Transformation matrices are used to create an open kinematic chain. The transformation matrix contains the robot's position and orientation information. A homogeneous transformation matrix is created for each joint. The degree of freedom (DOF) of the robot determines the number of these matrices created. Six degrees of freedom are sufficient to reach any point in three-dimensional space with any orientation. For this reason, industrial robots are generally produced with 6 DOF. Of course, there are robots with a DOF of less than six which can be used for simpler tasks. On the other hand, redundancy occurs in robots with more than six DOF. In this case, more than one joint is scanning the same area. This phenomenon is seen in special-purpose robots designed for special conditions.

One of the most important stages of robot studies is the kinematic model of robots. Many methods have been developed for this purpose. The solution to kinematic problems can be realized in two different spaces, three and four-dimensional. Since the kinematic model is extracted using transformation operators such as matrices or vectors in three-dimensional space, this method is called point transformation, and since linear vectors and transformation operators are used in four-dimensional space, this method is called linear transformation method (Aspragathos & Dimitros, 1998).

Robot Operating System

Robot Operating System is software that allows controlling robots. Although it is called an operating system, it is not an operating system. ROS is open-source interface software that creates human-robot communication.

ROS is widely used and preferred software in robots. The robot evaluates the data it receives from the environment via sensors and sends it as a command to the robot as a result. With the help of sensors in the controlled robot, data collected from the outside world such as sound and images are transferred to the computer via ROS. The data is processed according to the algorithm and usage area in the computer. The result of the processed data is sent back to the robot as a command. While performing these operations, it works in publishing/subscriber logic. The topic and messages are used to provide communication between the computer and the robot. In addition to being open-source, it is possible to use it with more than one programming language. In other words, it allows using different languages (Java, Lisp, C++, Python) on a robot.

With ROS, it is aimed for the user to develop software that fulfills the desired task by making minimum changes to the code and using it according to the needs. In addition, the user can quickly run the previously developed algorithms in the download-use format and thus make performance comparisons and impressions.

Robotics is a difficult and costly field to implement. For this reason, it may be important to test the application in a simulation environment before it is implemented in the real world. In the light of this importance, there are simulation environments that are fully

compatible with ROS. The most common of these simulations are Gazebo and Rviz environments. With the libraries and ready-made 2D and 3D objects in Gazebo, it offers a wide working area to the developers. Data packaged in ROS can be easily simulated and examined in 3D in the Gazebo environment.

While developing robot software, it can be used in interfaces such as Orocos, Carmen, Microsoft Robotics Studio, and YARP apart from ROS. The advantages of ROS over others can be summarized as follows.

- 1) It publishes a different issue (publish) in each transaction. Thus, error management can be made easier.
- 2) It is not dependent on a programming language. Among the languages it supports, there are many languages such as Java, C++, Python, and Lisp, which are among the most widely used programming languages. It is also platform-independent. Even a ROS node can be programmed in different languages according to need or teamwork. It comes to the fore in the realization of steps such as teamwork, remote work, or performance optimization by working integrated during compilation and study.
- 3) The software is open source. Being open-source code does not remain in a limited framework; it allows communicating with many people through the community. With the tools available in the community, studies can be developed in the desired direction.
- 4) Its modular structure is another advantage. While a problem in other robot control software causes the main node to stop, the node structure used here prevents this.

ROS is created from nodes as a system. These nodes communicate with each other and use the publish-subscribe messaging model. Units that can operate here are called nodes. For example, while one node performs the data collection process with the sensor from the outside world, another node performs the task of processing this data. Another node creates a result. The communication and operation of all these nodes with each other are carried out synchronously. To provide this, a server called ROS-Master creates the connection between the nodes. This Master server is an XML-RPC-based application.

Robotic Vision

With the emergence of deep learning, pattern recognition has become powerful tool that raises the direction of studies in the fields of machine learning and computer vision. Thanks to deep learning, many problems have been solved or reached better solution (Ucar & Kocer, 2022).. It has also become a research area and improvements have been made in various algorithms, data sets, and computational power. Such a powerful tool has attracted attention in all areas. Undoubtedly, robotic vision is among them. The deep learning algorithms used in robotic vision were transferred from computer vision. Robotic vision is the addition of action to the computer vision result. As shown

in Figure 2, if the output of the image inputs is data, it is computer vision. However, if the image inputs are transformed into action and a job is performed, robotic vision is realized. However, it is not possible to transfer every algorithm or method. Robotic vision and computer vision are separated because of the two main differs from each other on several issues.

- 1) Computer vision can generally be operated with hardware such as Graphical Processing Unit (GPU) that offers high computing power, while robotic vision is difficult to perform with such high hardware.
- 2) Robotic vision is run in real-time. For this reason, computer vision algorithms and methods can be transferred, which can meet the requirements of robotic vision such as real-time and limited computing power.

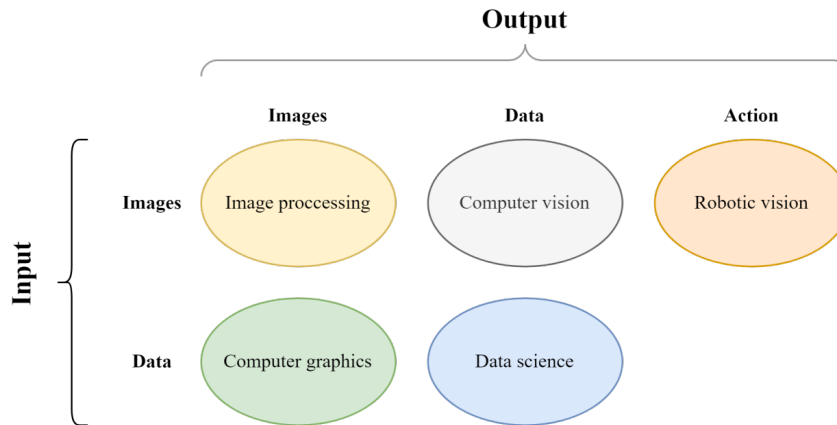


Figure 2. Position of robotic vision according to computer vision

An example of a robotic vision application is shown in Figure 3. General view of robotic vision and grasping mechanism is shown in Figure 3a. When the objects on the conveyor belt (Figure 3b) will be separated with the robot arm, the objects are viewed with a camera as in Figure 3c. By running the deep network on the displayed objects, necessary information for gripping such as detection, location, and speed of the objects is obtained. According to the information obtained, the robot arm positions itself as in Figure 3d. Then, the robot arm waits for the objects to come to the grip area as in Figure 3e. When the robot arm realizes the grip, it separates the desired object (Figure 3f). Since all these processes have to be carried out in a short time, the need for fast-running and moving systems is huge.

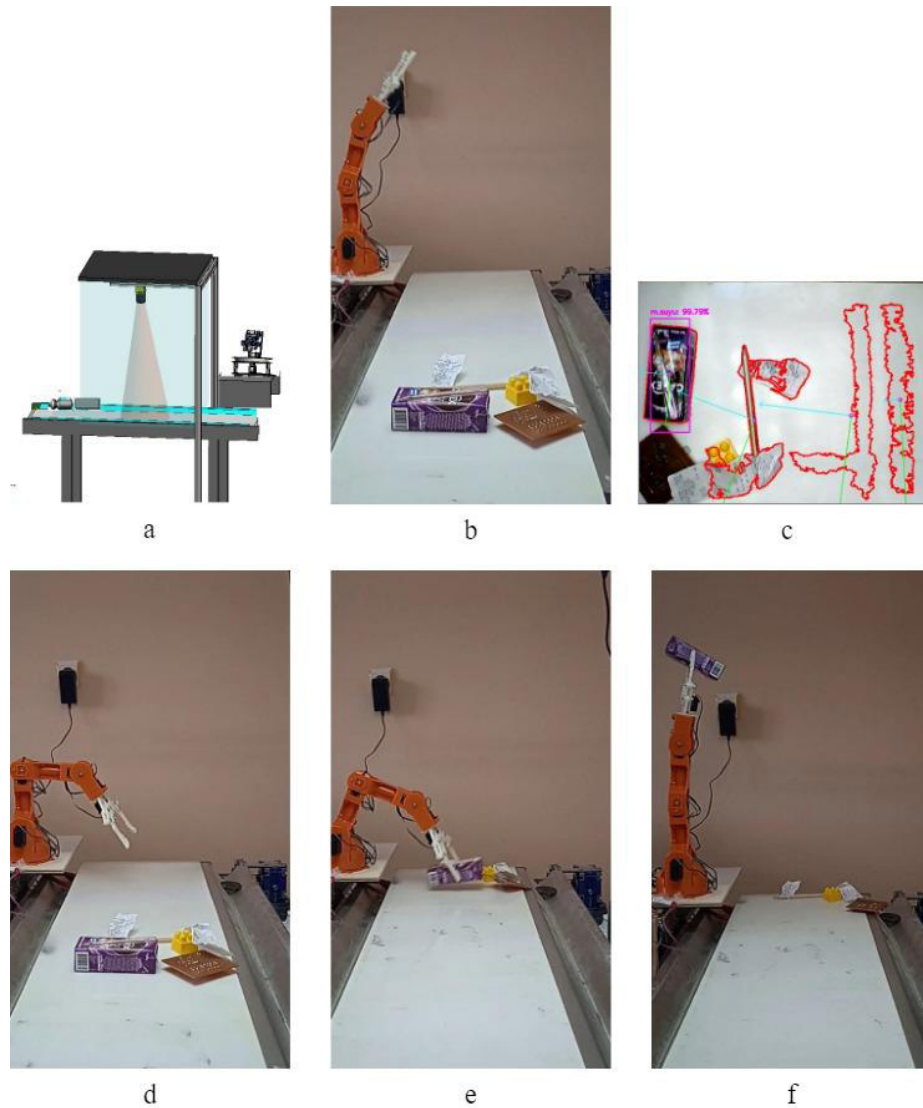


Figure 3. a) General view of robotic vision & grasping mechanism, b) Robot arm and objects to be separated, c) Viewing objects and obtaining deep network related data, d) Positioning of the robot arm according to the deep learning result, e) The robot arm grasps the desired object, f) Separation of the grasped object

Deep Learning

Deep learning seeks solutions using a more complex architecture for problems where artificial intelligence algorithms are insufficient. For this reason, it needs higher computational power and data set. In the early years, it took some time to realize its importance and power, since it was not possible to meet these needs. Many different architectures have been developed as it has become easier to train and use.

Convolutional Neural Networks

CNN is an architecture of DNN. This structure, inspired by the visual system of mammals, has local connectivity, pooling, shared weight, and the use of many layers (Bengio & LeCun, 2007). The large numbers of hidden layers that make up CNN are used in 3 different tasks. Convolutional layers with linear filtering are the layers where the features are obtained. The non-linearity of the network is gained with the activation function

applied after the convolution layer. A pooling layer is used between the convolution layers. Pooling layers serve to reduce the data size to reduce the processing load. The use of the pooling layer should be careful, as there will be data loss along with the data size reduction. Maximum and average pooling are the most commonly used pooling layers. After multiple layers of convolution and pooling, an input-based decision is required. Decision-making is done in the fully connected layer. The operations of these layers are shown in Figure 4.

The convolution, pooling, and fully connected layers that make up a CNN, the number and size of filters vary according to the application. However, since there must be at least one connection for each data in each layer, it needs quite high computational power. However, since most operations are done locally in CNN, it is suitable to work with GPUs.

Fully connected layers are an important layer for converting 2D data to 1D. This layer is similar to the classical ANN. Also here the final classification is carried out.

CNNs can be created in different structures by using the different number of layers, filter size, and number, dropout, pooling, and activation functions. Many architectures have been created according to these differences and attract attention with their performance (AlexNet (Krizhevsky et al., 2012); VGG (Simonyan & Zisserman, 2014); GoogleNet (Szegedy et al., 2015); ResNet (He et al., 2016); DenseNet (Huang et al., 2017)).

Although CNN is successful in classification, it cannot produce an output for object tracking, recognition, and tracking tasks. In such tasks, the target usually covers a small area in the images and this area needs to be detected. The window scrolling method can be used for this task, but its slow operation makes it impossible to use for robotic tasks. To eliminate this disadvantage, it is suggested to use region proposals containing objects. Fast RCNN (Girshick, 2015) is one of these methods. In the light of later studies, SSD (Liu et al., 2016) and YOLO (Redmon et al., 2016) emerged. YOLO has been an algorithm that has been studied a lot. It is a very effective method as it can perform object classification and location detection with a single CNN. It also allows trade-offs between accuracy and speed. The most striking feature is that it offers high speed and accuracy even with low hardware, thanks to different architectures and developments.

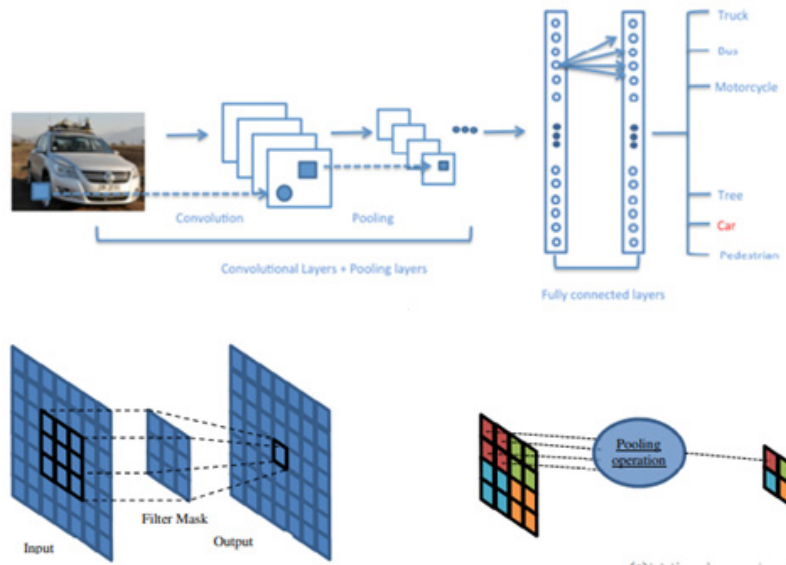


Figure 4. The operations of CNN layers (Guo et al., 2016)

Fully Convolutional Networks

The difference between CNN is that there is no fully connected layer in Fully CN. For this reason, the output in the last convolution layer is the network output. In these networks, the input and output also consist of 2D or 3D tensors. Among the most common uses are dimensionality reductions, regression, or semantic segmentation.

One type in which FCNs are used is Auto Encoders (Hinton & Salakhutdinov, 2006). An encoder and decoder are created using two FCNs. The encoder reduces the size of the input image and creates a tensor output. In the decoder, this tensor regenerates the image by performing inverse operations. Auto encoder's training is also carried out for this purpose. For this reason, it can be used in noise removal and segmentation tasks.

Deep Reinforcement Learning

Reinforcement Learning consists of a set of techniques that learn by rewarding agents cumulatively for their success in their tasks. Here, agents interact with their environment to take an action and receive rewards (Li, 2017).

Deep Learning in Robotic Vision

Robotic vision studies using deep learning are divided into three different categories (Ruiz-del-Solar et al., 2018): object detection and categorization, object grasping and manipulation, scene representation, and classification. Although many robotic vision studies have been carried out in these areas, it is not known whether they can meet the real robotic vision requirements as some of them are carried out in simulation environments.

- **Object Detection and Categorization:** One of the most fundamental steps of robotic applications is object recognition and classification. Thus, robots can interact with the real world. As a result of the interaction, the robots can detect the surrounding targets and, if necessary, avoid obstacles such as walls and

foreign objects that will prevent them from working. Of course, this contribution of deep learning to robots becomes limited by the real-time requirement. As a result, applications are carried out around task-based methods. With task-based applications, it is aimed to work in real-time by using the most useful, fast, and few features of algorithms with a limited number of objects. In addition, some applications have increased the speed by using smaller deep networks.

- **Scene Representation and Classification:** In recent years, mobile robots have become quite common. With these robots, which have many different tasks, each scene is different and can be important in terms of their duties. Even outside the task, robots can understand the movement route and location information through images. For this, the features that were previously extracted manually have become easy with the help of deep learning.
- **Object Grasping and Manipulation:** Grasping is a very important task in robotic applications. Both gripper and object detection are areas where deep learning is frequently used. In this context, tasks such as grasping, manipulation, and visual servoing are performed.
- Although the task of grasping objects is very easy for humans, the situation for robots is quite complex. With the help of deep learning, robots can gain the ability to grasp objects in a fixed shape. However, as the shapes, sizes, positions, and directions of objects change, robots face a difficult situation. In such cases, robots undergo long-term training with trial-and-error methods.

Deep Learning in Robotic Vision

Most jobs involving robots also involve object grasping. For this reason, a gripper is mounted on the end of the robots. Grippers are mechanical, pneumatic, or electrical equipment used to carry or hold a part. The selection of this critical part can be chosen according to the task to be performed by the robot. The most widely used robotic grippers are; 1) Parallel Motion Two-Jaw Gripper, 2) Three-Jaw Gripper, 3) Bellows Gripper, 4) Collet and Expanding Mandrel Grippers, 5) O-Ring Grippers, 6) Needle Grippers, 7) Multi-Finger and Adaptive Grippers, 8) Vacuum Cup or Vacuum Cup Arrays, Electromagnets, and Electrostatic Force Grippers.

Deep Reinforcement Learning

A frequently preferred area of deep learning is grasping. Grasp can be defined as the partial or complete closure of an object with an end effector. A gripping problem can be solved by correctly positioning the robot. For this, it is necessary to know the features of the object to be grasped, such as shape, size, and location. According to these features, possible gripper configurations should also be investigated (Bohg et al., 2013; Kumra & Kanan, 2017).

Robots need as much information as possible about the targets they need to grasp.

Presenting this information by people who are experts in the subject is both a time-consuming and a very difficult approach.

For humans, grasping is an unthinking and automatic action. However, the same is not true for robots. Comprehension of objects is a very challenging problem due to factors such as an unlimited variety of objects, different physical properties of objects, and different viewing angles. All these difficulties must be overcome to achieve successful grasping. When we want to grasp an object, we see the object and determine its position. Then we reach out and hold the object. The same procedures apply to robots. However, since robots are not as smart as humans, they cannot practice grasping that much. In addition, robotic grippers can even have only two fingers. After detecting the target on the images, the robots obtain their position in the image. It should estimate the location in the real world based on the location in the image. It must position itself and calculate grip points by comparing its position with the position in the real world. Then he should reach and grasp the object without being hindered by the surrounding obstacles.

References

- Aspragathos, N. A. & Dimitros, J. K. (1998). A comparative study of three methods for robot kinematics. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(2), 135–145.
- Bengio, Y. & LeCun, Y. (2007). Scaling learning algorithms towards AI. *Large-Scale Kernel Machines*, 34(5), 1–41.
- Bohg, J., Morales, A., Asfour, T. & Kragic, D. (2013). Data-driven grasp synthesis—a survey. *IEEE Transactions on Robotics*, 30(2), 289–309.
- Girshick, R. (2015). Fast r-cnn. *Proceedings of the IEEE International Conference on Computer Vision*, 1440–1448.
- Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S. & Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, 187, 27–48.
- He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.
- Hinton, G. E. & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504–507.
- Huang, G., Liu, Z., Van Der Maaten, L. & Weinberger, K. Q. (2017). Densely connected convolutional networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4700–4708.
- Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25.
- Kumra, S. & Kanan, C. (2017). Robotic grasp detection using deep convolutional neural networks. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 769–776.

- Li, Y. (2017). Deep reinforcement learning: An overview. *ArXiv Preprint ArXiv:1701.07274*.
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. & Berg, A. C. (2016). Ssd: Single shot multibox detector. *European Conference on Computer Vision*, 21–37.
- Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 779–788.
- Ruiz-del-Solar, J., Loncomilla, P. & Soto, N. (2018). *A Survey on Deep Learning Methods for Robot Vision*. 1–43. <http://arxiv.org/abs/1803.10862>
- Simonyan, K. & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *ArXiv Preprint ArXiv:1409.1556*.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. & Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9.
- Ucr, K., & Kocer, H. E. (2022). Determination of Angular Status and Dimensional Properties of Objects for Grasping with Robot Arm. *IEEE Latin America Transactions*, 100(XXX). Retrieved from <https://latamt.ieee9.org/index.php/transactions/article/view/6999>

About the Authors

Kursad UCAR received the Master degree from the Department of Electric–Electronic Engineering, Institute of Science, Selcuk University, Konya, Türkiye (2018). He continues his Ph. at the same university. He has been a research assistant at Selçuk University since 2016. His research interests include image processing, robotics, automation and control, and machine learning.

E-mail: kucar@selcuk.edu.tr, **ORCID:** 0000-0001-5521-2447

Hasan Erdinc KOCER received the Ph.D. degree from the Department of Electric–Electronic Engineering, Institute of Science, Selcuk University, Konya, Türkiye (2007), where he has been an associate professor since 2018. His research interests are in machine vision and pattern recognition, industrial automation on camera, medical image processing, and automation and control.

E-mail: ekocer@selcuk.edu.tr, **ORCID:** 0000-0002-0799-2140

Similarity Index

The similarity index obtained from the plagiarism software for this book chapter is 3%

To Cite This Chapter:

Ucar K, Kocer H. E. (2022). Robotic Vision with Deep Learning. In S. Kocer & O. Dundar (Eds.), *Current Studies in Basic Sciences Engineering and Technology* (pp. 1–11). ISRES Publishing.