sciendo

DOI 10.2478/ama-2019-0028

acta mechanica et automatica, vol.13 no.3 (2019)

PICK-AND-PLACE TASK IMPLEMENTATION USING VISUAL OPEN-LOOP CONTROL

Paweł KOŁOSOWSKI,[*] Adam WOLNIAKOWSKI,[**] Mariusz BOGDAN[*]

[*]Faculty of Mechanical Engineering, Bialystok University of Technology, ul. Wiejska 45C, Białystok 15-351, Poland
[**]Faculty of Electrical Engineering, Bialystok University of Technology, ul. Wiejska 45D, Białystok 15-351, Poland

pawelkolosowski0@gmail.com, a.wolniakowski@pb.edu.pl, m.bogdan@pb.edu.pl

received 10 July 2019, revised 30 October 2019, accepted 31 October 2019

Abstract: In the ever increasing number of robotic system applications in the industry, the robust and fast visual recognition and pose estimation of workpieces are of utmost importance. One of the ubiquitous tasks in industrial settings is the pick-and-place task where the object recognition is often important. In this paper, we present a new implementation of a work-piece sorting system using a template matching method for recognizing and estimating the position of planar workpieces with sparse visual features. The proposed framework is able to distinguish between the types of objects presented by the user and control a serial manipulator equipped with parallel finger gripper to grasp and sort them automatically. The system is furthermore enhanced with a feature that optimizes the visual processing time by automatically adjusting the template scales. We test the proposed system in a real-world setup equipped with a UR5 manipulator and provide experimental results documenting the performance of our approach.

Key words: Grasping, pick-and-place, object recognition, template matching, visual servoing

1. INTRODUCTION

A typical configuration of a pick-and-place task is presented in Fig. 1. A vision system is used to recognize and estimate the pose of objects located in the workcell. This information is subsequently used by the user application to control the robotic manipulator in order to grasp the objects and place them in the designated target areas. Such a setup typically requires accurate calibration.

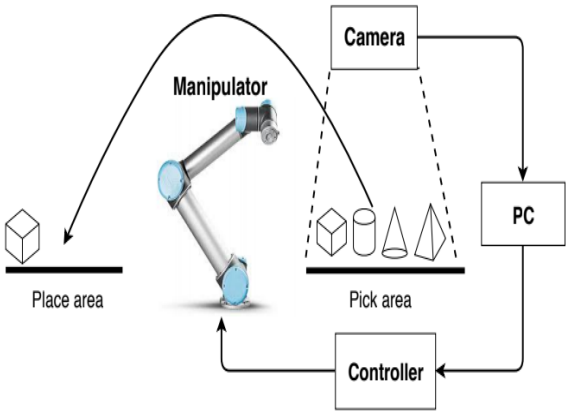


Fig. 1. Typical pick-and-place task

In this paper, we present an implementation of a 2D object recognition and sorting task using a robot manipulator and a vision system.

The successful execution of the pick-and-place task, where multiple types of objects are present, requires solving an object recognition problem. In recent years, the problem of sorting of objects of different shape, size and colour was extensively researched in the literature (Saxena et al., 2007; Kumar et al., 2014; Willaume et al., 2016; Nalini and Gondkar, 2017).

The research is often focused on the problems encountered in successful implementation of the system. These problems in the context of the pick-and-place task revolve around two most important aspects: quick and correct visual recognition of the object, and a stable grasp (Amagai and Takase, 2017).

On the vision system side, the current trend in the industry is to replace the mechanically forced object position with 'smart' recognition by the use of a vision based object recognition (Steger et al., 2017). The performance of the vision system is chiefly impacted by the object complexity, the number of objects present in the scene, the lightning conditions and the calibration quality. Solving the generalized vision problem is often not feasible due to practical considerations. Therefore, research on visual based object recognition is usually restricted to particular classes of the objects, for example, geometrical characteristics, colour or size. Despite significant progress in the recent years, still there are issues that are not handled by the vision systems very well. These include, for example, segmentation and recognition of non-rigid objects, where the observed shape is allowed to vary and there is lightning imbalance due to shadows, reflections and the object texture (Hagelskjær et al., 2017).

Object recognition is an interesting research area, very important in robotic applications. Multiple methods are currently known. One of the earliest approaches involved matching the features of the known object model (Grimson, 1990; Ellgammal, 2005; Sibiryakov, 2008) or shape primitives based recognition (Krivic and Solina, 2004; Nieuwenhuisen et al., 2012,). Currently, a very popular trend is to use machine learning algorithms (Cabre et al., 2013). Most of these approaches rely on the feature extrac-

DOI 10.2478/ama-2019-0028

tion and matching methods, such as SIFT (Lowe, 2004; Collet, et al. 2011) or SURF (Bay et al., 2008). SIFT algorithm is robust in respect to cluttered background, perspective shift, lightning conditions, scaling and rotation, but requires precise image segmentation. SURF is considered to have better performance. Still, both of these methods suffer issues due to uniform colour scheme and smooth contours of the recognized objects.

The vision systems used in industry and research can be classified into different categories. One of such classifications involves the distinction on the camera placement, which can be fixed in the workcell or attached to the robot manipulator (Chaumetter and Hutchinson, 2007; Corke, 2001; Flandin et al., 2000). The latter case has recently resulted in the emergence of the visual servoing methods (Pessoa et al., 2018; Huang and Xu, 2018; Amini and Banitsas, 2019). Both types of the vision systems offer advantages and shortcomings. The fixed camera system offers higher precision due to the stability of the calibration, while suffering from the limitation to the observed area and just one perspective. The moving camera allows for the observation of the scene from multiple vantage points, while having lower precision at the same time. Typically, the fixed camera vision systems are still used in most scenarios. The overview of our proposed system is presented in Fig. 2. Our system is divided into two parts: the offline template scale optimization process and the online process executed during the pick-and-place task realization. Both branches of the system share the first two steps: the image acquisition using the vision system and the image rectification, where the distortions are removed from the acquired image. This is performed based on the intrinsic camera calibration.

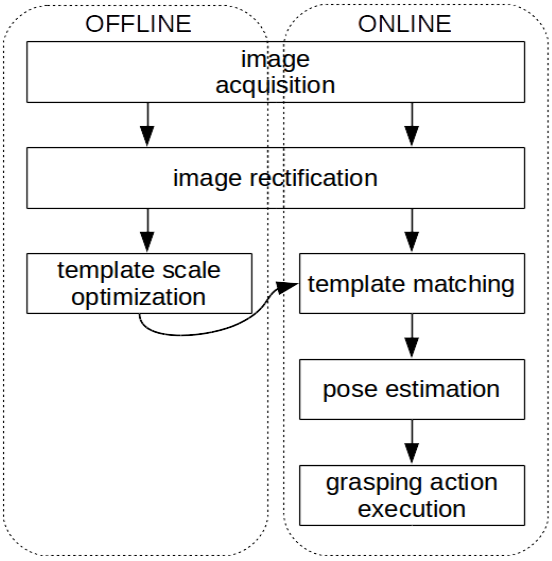


Fig. 2. Proposed system implementation workflow diagram

The offline branch of the system is responsible for preparing the templates for the template matching algorithm. Poor extrinsic camera-to-robot calibration – or even lack thereof – necessitates the assumption of multiple scales of templates, which greatly increases the computation time in the online phase. Our proposed solution uses this offline step to select the template scale that matches the object scale in the scene, thereby improving the vision system processing rate.

The online branch of our system is concerned with controlling the manipulator in order to perform the pick-and-place task based on the information on the recognized object pose obtained through the vision system. We have subsequently tested our proposed approach in an industrial-based real experiment. We provide quantitative results on the performance of our template matching system, thereby proving the feasibility of its implementation in automizing pick-and-place tasks.

2. METHODS

In this section, we present the detailed description of the individual components of our template matching system.

2.1. Template matching

Template matching is an image recognition technique based on comparing the predefined image (template) against the source image, in order to find areas with highest similarity. In this method, the template is sled over the source image and the ratio of similarity is computed, which describes how well the template matches the image in specific position, according to the specified method. There are several comparison methods; we have chosen normalized correlation coefficient, which is described in the equation (1).

$$R(x,y) = \frac{\sum_{x',y'} T'(x',y') * I'(x+x',y+y')}{\sqrt{\sum_{x',y'} T'(x'y')^2 * \sum_{x',y'} I'(x+x',y+y')^2}} \quad (1)$$

where: I is image, T - template, R - matrix of matching results that contains values between 0 and 1, where 1 means perfect match and 0 means a complete mismatch.

Template matching is a computationally complex algorithm the execution time of which increases with template and source image resolution. To speed-up the process, we used templates with lowest possible resolution, which maintained the geometry of the objects (i.e., lowest template resolution for which the different objects were still distinguished). We define the ratio between the actually used template width and this normalized width as the template scale.

Owing to the fact that template matching algorithm is not scale and rotation invariant, there is a necessity to generate templates in multiple scales and rotation angles. Scaling is usually performed by generating image pyramid with downsamples or upsamples of the given template. However, we have implemented a less time-consuming approach by using only one optimal preprocessed scale for every template (see 2.2). In order to define object rotation, every template has been rotated in range from 0° to 360° every 2°.

The grasping poses can be defined for the objects by the user as the transform between the object's CoG and the desired grasp pose. In case the grasp pose is not defined, an identity transform is assumed by default.

2.2. Template scale optimization

Since the system is designed to handle typical pick-and-place actions with the objects placed in the constant distance to the camera, no variation of the object image size is expected. Still, without prior camera-to-table pose calibration, the apparent size of the image is unknown. In order to escape the necessity of per-

forming the time-consuming extrinsic calibration, it is possible to find the optimal template scale in the template pre-processing phase.

In order to find a scale that provides the highest recognition efficiency for the given template, we have taken 20 images of scenes with single object placed in various orientations. For each scene, template has been generated in wide range of scales and compared against the object segmented from the scene. The optimal scale for the template has been designated as an average of those template scales, which gave the highest correlation coefficient value for every image. The results of this process in our experiment are presented in Section 3.3.

2.3. Experimental setup

A laboratory setup at BUT was used to perform the real world experimental testing of the proposed method. The setup consists of a UR5 manipulator mounted in the aluminium frame workcell (see Fig. 3). The objects were placed on the surface of the table, where areas were designated for the object picking and placement. The camera used was 1280 x 1024 [px] uEye Camera.

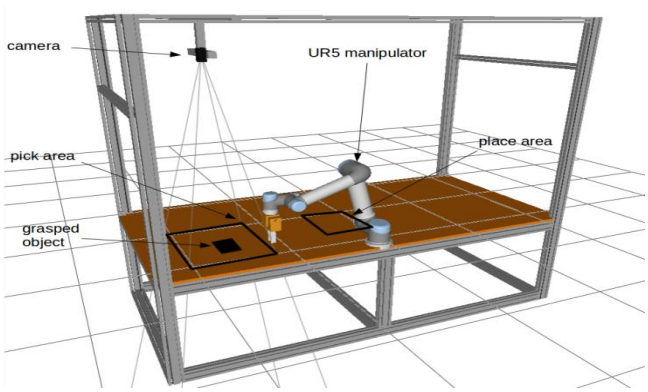


Fig. 3. The experimental setup modelled in RobWorkStudio

The UR5 manipulator used in the setup was equipped with a Schunk EVG55 electric gripper. The manipulator is capable of reaching the objects placed in the designated area and lifting up to 5 kg of weight. The experimental setup configuration is presented in Fig. 4. The ROS (Robot Operating System) package controlling the hardware is running on the laboratory workstation. The vision algorithm uses the OpenCV library to connect with the camera and process the image. The application uses ROS topics and services to control the robot and the gripper.

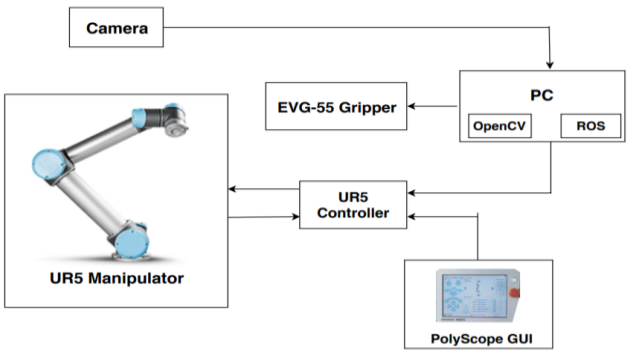


Fig. 4. The experimental setup configuration

3. RESULTS

In order to test the performance of our template matching vision system in the real setup, we have executed a pick-and-place experiment. We have tested our recognition algorithm with high success rate for both simple and complex object geometries. For the experiment, we have selected three objects: crank (object 1), bracket (object 2) and spacer (object 3). These objects are presented in Fig. 5. The objects were selected such as to have sparse visual features. The two objects (crank and bracket) were chosen such as to test the capability of the system to distinguish between similar shapes. The third object (spacer) was provided as the control.

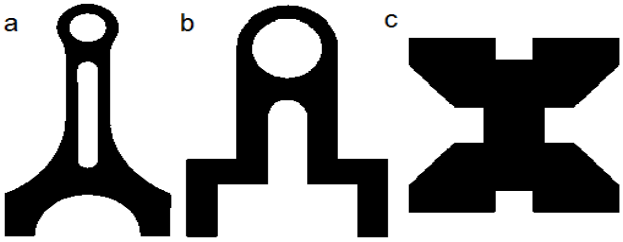


Fig. 5. The objects selected for the experiment: a) crank; b) bracket; c) spacer

The experiment was performed as described below.

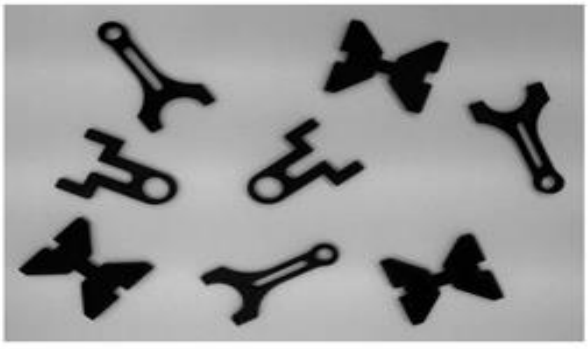
3.1. Camera calibration

Firstly, the intrinsic camera calibration was performed in order to rectify the distortion of the image. The calibration was done by presenting a chessboard calibration pattern to the sensor in 130 poses and extracting the intrinsic camera matrix. This calibration was done using the OpenCV software package (Bradski, 2000). Next, the intensity analysis was performed in order to account for the lightning influence on the threshold for image segmentation (see Fig. 6). Sample images were taken, and the intensity histogram generated, from which the segmentation threshold was derived. We used the following formula to calculate the segmentation threshold (2).

$$\mathrm{I}_{thr} = 0.5(\mathrm{I}_{obj\_max} + \mathrm{I}_{bg\_max}) \quad (2)$$

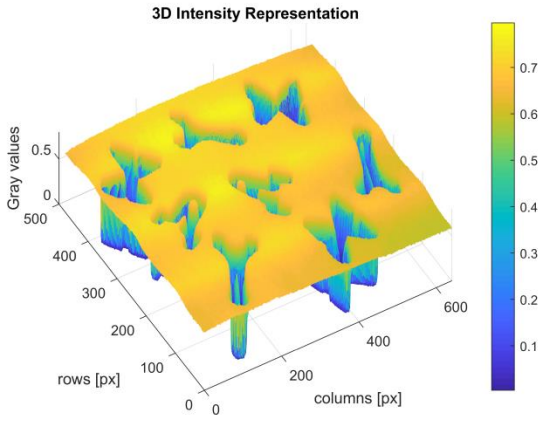
where: I_{thr} – segmentation threshold; I_{obj_max} – object histogram peak; I_{bg_max} – background histogram peak (see Fig. 6c).

a)



DOI 10.2478/ama-2019-0028

b)



c)

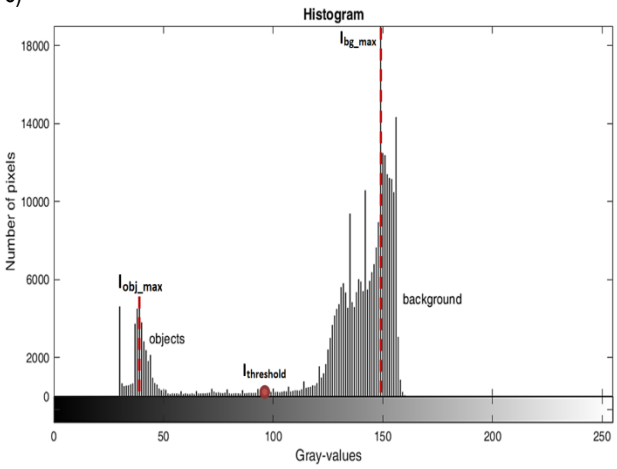


Fig. 6. Image intensity analysis: a) Sample camera image; b) 3D image intensity representation; c) Sample image intensity histogram

3.2. Image-to-robot calibration

In order to find the correspondence between the image pixel coordinates and the robot workspace coordinates, it was necessary to find the transform between the two. Since the designated grasping area was placed on the same plane as the robot base, the process was simplified to the problem of finding an affine transformation between the image space and the base plane. In order to find the transformation, we measured a set of positions uniformly distributed in the camera field of view obtaining both the pixel coordinates $[u, v]$ and the robot TCP $[x, y, \theta]$ coordinates. We then obtained the transformation matrix T using least square optimization (3):

$$\mathbf{T} = \begin{bmatrix} x_1 & y_1 & \theta_1 \\ x_2 & y_2 & \theta_2 \\ \vdots & \vdots & \vdots \\ x_n & y_n & \theta_n \end{bmatrix} \begin{bmatrix} u_1 & v_1 & 1 \\ u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots \\ u_n & v_n & 1 \end{bmatrix}^{+} \tag{3}$$

where: u_1 and v_1 are the image pixel coordinates, while x_1, y_1 and θ_1 are the robot TCP coordinates and *n* is the number of the calibration samples taken.

The obtained transformation matrix **T** was then used to find the robot TCP target coordinates as the function of the image coordinates (4).

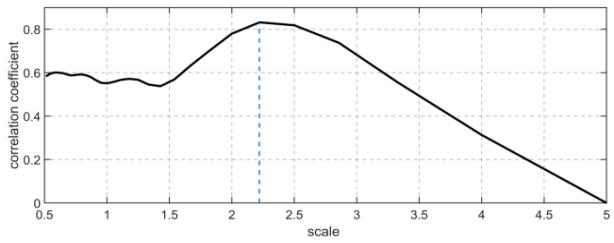
$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \mathbf{T} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{4}$$

In our experiment, we used 24 calibration samples and obtained the 0.92 [mm] RMSE value for the coordinate transform between the image and the robot frame coordinates.

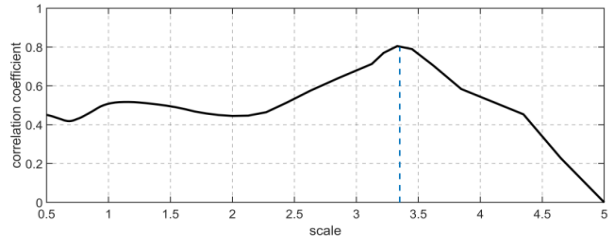
3.3. Optimal Template Scaling

The next step of our experiment was to find the optimal template scales for the three selected objects. For each of the objects, a set of scaled templates was generated and tested against the baseline camera image. For each of these scales, the correlation coefficient was computed and the optimal scaling factor was found. The relationship between the template scale and the matching score for the analysed objects is presented in Fig. 7.

a)



b)



c)

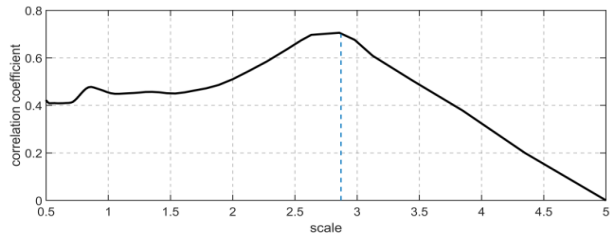


Fig. 7. The correlation coefficient as the function of template scale for three considered objects: a) Object 1 – crank; b) Object 2 – bracket; c) Object 3 – spacer

The following template sizes were chosen based on these results: 2.22; 3.40; 2,83 for object 1 (crank) 20 x 34 [px] template, 2 (bracket) 30 x 39 [px] template and 3 (spacer) 30 x 36 [px] template respectively.

The processing time for the object recognition also depends on the selected template resolution. The average recognition time for different template resolutions are shown in Fig. 8. In this work, we select the template scale based on the matching score, but it is conceivable to introduce a trade-off when faster on-line processing is required.

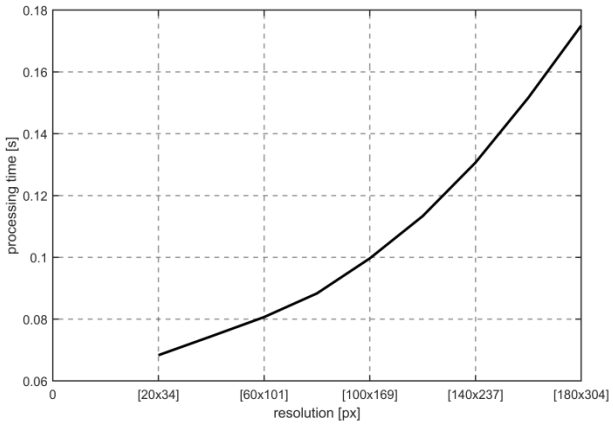


Fig. 8. Average single object recognition time as the function of template resolution

3.4. Manipulator control

In order to perform pick-and-place task, online-generated manipulator trajectory has been planned with the usage of recognized objects CoGs, rotations angles and desired grasp poses. Robot control was realized through point-to-point motions' implementation provided by the manufacturer. Pick-and-place actions were realized through linear movement in the Cartesian space, while simple joint-space point-to-point movements were used for the transfer of the pieces. The UR5 robot achieves +/- 0.1 [mm] repeatability according to the manufacturer.

3.5. Pick-and-place task

In the grasping task, the system was used to recognize and estimate the pose of 6 objects (2 of each kind) placed in the designated picking area as shown in Fig. 3. The camera image was captured and upon processing, the objects were grasped and placed in the target area. This action was repeated 50 times, and thus, 300 recognitions were performed (6 objects viewed 50 times). In this experiment, each object was expected to be recognized 100 times. During the experiment, we analysed whether the objects were correctly recognized and if they were placed in the proper target areas. The data collected during this experiment is presented in the Tab. 1 below.

Tab. 1. The confusion matrix of the cumulated experimental results

	Recognized as:			
	Object 1	**Object 2**	**Object 3**	**Not recognized**
Object 1	100	0	0	0
Object 2	0	100	0	0
Object 3	0	0	98	2

99.33% was achieved, where only the third type of object (the spacer) was not recognized in two cases. The results gathered in the confusion matrix suggest that the proposed method performs well. We have additionally recorded the vision processing time in order to track the system performance. We have achieved 0.21 [s] average recognition time per scene.

4. CONCLUSIONS

In this paper, we have presented a new implementation of the template matching algorithm applied to the solving of an industrial based pick-and-place task, where the grasped objects are characterized by few visual features. This property of the objects often makes the task impossible to solve using sophisticated vision algorithms, which rely on multiple and distinct features. We have proposed and described a two part system, in which an offline template scale optimization is introduced in order to improve the online vision system performance. The introduced system allows the user to present the object templates and define the grasping pose using an intuitive GUI.

We have tested the proposed method in a real-world setting using a robotic setup equipped with an UR5 manipulator. Three object shapes were selected for the experiment in order to test the ability of the system to distinguish distinct geometric shapes. The experiment provided the information on the success rate of the object recognition, the rate of successful grasping and the achieved processing time. A 99.33% success rate was achieved by the system, where only one kind of an object was not recognized twice in 100 samples. The average processing time achieved was 0.21 [s] per scene, which is sufficient for an online implementation. The results obtained in the experiment support the feasibility of our proposed approach. Still, the system setup requires some cost in terms of the need for calibration. The future work should be focused on reducing the setup time required, such as performing automatic camera-to-robot calibration.

REFERENCES

1. **Amagai A., Takase K.** (2001), Implementation of dynamic manipulation with visual feedback and its application to pick and place task, *Proceedings of the 2001 IEEE International Symposium on Assembly and Task Planning Assembly and Disassembly in the Twenty-first Century,* 344–350.
2. **Amini A., Banitsas K.** (2019), Using Kinect v2 to Control a Laser Visual Cue System to Improve the Mobility during Freezing of Gait in Parkinson's Disease, *Journal of Healthcare Engineering,* art. no. 3845462.
3. **Bay H., Tuytelaars T., Gool L.V.** (2008), Speeded-up robust features (SURF), *Computer Vision and Image Understanding*, 110, 346–359.
4. **Bradski G.** (2000), The OpenCV Library, *Dr. Dobb's Journal of Software Tools*.
5. **Cabre T.P., Cairol M.T., Calafell D.F., Ribes M.T., Roca J.P.** (2013), Project-Based Learning Example: Controlling an Educational Robotic Arm with Computer Vision, *Tecnologias del Aprendizaje, IEEE Revista Iberoamericana de, 8,* 135–142.
6. **Chaumette F., Hutchinson S.** (2007), Visual Servo Control, Part II: Advanced Approaches*, IEEE Robotics and Automation Magazine,* 14(1), 109–118.
7. **Collet A., Martinez M., Srinivasa S.S.** (2011), The MOPED framework: object recognition and pose estimation for manipulation, *The International Journal of Robotics Research,* 30, 1–23.
8. **Corke P.** (2001), Robotics, Vision and Control: Fundamental Algorithms in MATLAB, *Springer*.
9. **Ellekilde L.-P., Jorgensen J. A.** (2010), Robwork: A flexible toolbox for robotics research and education, *Proceedings of the 41st International Symposium on Robotics and 6th German Conference*, 1–7.
10. **Ellgammal A.** (2005), Object Detection and Recognition, *Rutgers University*.

DOI 10.2478/ama-2019-0028

11. **Flandin G., Chaumette F., Marchand E**. (2000), Eye-in-hand/eye-to-hand co-operation for visual servoing, *Proceedings of the IEEE International Conference on Robotics and Automation*, 3, 2741–2746.
12. **Grimson W.** (1990), *Object recognition by computer: the role of geometric constraints*, MIT Press.
13. **Hagelskjær F., Krüger N., Buch A.G.** (2016), Does Vision Work Well Enough for Industry?, *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications.*
14. **Huang W., Xu H.** (2018), Development of six-DOF welding robot with machine vision, *Modern Physics Letters B,* 32, 34–36.
15. **Krivic J., Solina F.** (2004), Part-level object recognition using superquadrics, *Computer Vision and Image Understanding,* 95, 105–126.
16. **Kumar R., Kumar S., Lal S., Chand P.** (2014), Object detection and recognition for a pick and place robot, *Computer Science and Engineering 2014 Asia-Pacific World Congress,* 1–7.
17. **Lowe D.G.** (2004), Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision,* 60, 91–110.
18. **Nalini K.M., Gondkar R.R.** (2017), Robotic recognition for unstructured 2-D parts to pick and place objects, *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology,* 1478–1482.
19. **Nieuwenhuisen M., Stückler J., Berner A., Klein R., Behnke S.** (2012), Shape-primitive based object recognition and grasping, *Proceedings of the 7th German Conference on Robotics*.
20. **Pessoa R., Barbosa W., McLoughlin J., Kokaram A.** (2018), Visual Servo Control of a Micro Quad-copter as a Teaching Platform for Engineering, *29th Irish Signals and Systems Conference,* art. no. 8585355.
21. **Saxena A., Driemeyer J., Ng A.Y.** (2007), Robotic Grasping of novel Objects using Vision, *The International Journal of Robotics Research*, 27(2), 157–173.
22. **Sibiryakov A.** (2008), Statistical template matching under geometric transformations, *Discrete Geometry for Computer Imaginary, Proceedings of the 14th IAPR International Conference,* 225–237.
23. **Steger C., Ulrich M., Wiedmann C.** (2017), Machine Vision Algorithms and Applications, *Wiley-VCH.*
24. **Willaume P., Parrend P., Gancel E., Deruyver A.** (2016), The graph matching optimization methodology for thin object recognition in pick and place tasks, *2016 IEEE Symposium Series on Computational Intelligence*, 1–8.