Implementation of Object Detection and Recognition Algorithms on a Robotic Arm Platform Using Raspberry Pi

Cagri KAYMAK
Mechatronics Engineering Department
Firat University
Elazig, Turkey
ckaymak@firat.edu.tr

Aysegul UCAR
Mechatronics Engineering Department
Firat University
Elazig, Turkey
agulucar@firat.edu.tr

***Abstract*— In this paper, it is aimed to implement object detection and recognition algorithms for a robotic arm platform. With these algorithms, the objects that are desired to be grasped by the gripper of the robotic arm are recognized and located. In the experimental setup that established, OWI–535 robotic arm with 4 DOF and a gripper, which is similar to the robotic arms used in the industry, is preferred. Local feature-based algorithms such as SIFT, SURF, FAST, and ORB are used on the images which are captured via the camera to detect and recognize the target object to be grasped by the gripper of the robotic arm. These algorithms are implemented in the software for object recognition and localization, which is written in C++ programming language using OpenCV library and the software runs on the Raspberry Pi embedded Linux platform. In the experimental studies, the performance of the features which are extracted with the SIFT, SURF, FAST, and ORB algorithms are compared. This study, which is first implemented with OWI–535 robotic arm, shows that the local feature-based algorithms are suitable for education and industrial applications.**

***Index Terms*— Object detection and recognition, Local feature-based algorithms, OWI–535 robotic arm, Raspberry Pi, OpenCV.**

I. INTRODUCTION

Computer vision is an important sensing technology in the robotics area because it has potential applications in many industrial processes. Object recognition, which is one of the popular applications of the computer vision, is the process of finding and classifying objects that have a common feature or relationships with each other through various processes.

Object detection and recognition applications are generally made using appearance-based or local feature-based approaches, depending on the purpose of use. The local feature-based approach is used when appearance-based approaches are not enough. In cases where illumination changes or objects are partially occluded by another object, local feature-based approaches are usually preferred. Local features can be expressed as specific regions containing information about objects. The feature vectors obtained from this approach are descriptors such as distances to the center, curvatures of curves and corners [1]. Through these features, objects can be defined independently from the whole.

Due to the difficulties encountered in object recognition applications, object recognition algorithms have an extensive literature. With the local feature-based approach, image matching studies are started with the corner detector of Morevec [2]. This detector, developed by Morevec, has been developed by Harris and Stephans [3] to give better results on the details of the image and the edges near each other. In [4], Harris used the developed detector for motion tracking and reconstruction of 3D structures. Schmid and Mohr [5] created local features using Harris detector with their chosen points. In addition, they have achieved successful results by defining vectors that are independent of orientation. With the development of local feature matching, they have also successfully performed feature search within a wide range of image databases. There are many studies that use appearance-based object recognition approaches [6-9].

The common problem encountered with object recognition algorithms, which use the corner points to construct the local feature vectors, is that they must be successful only when working on a single scale. When working at different scales, it is not possible to achieve independence from the scale because the points determined in each scale are in different positions. With Scale-Invariant Feature Transform (SIFT) algorithm developed by Lowe, the local features of the object were extracted independently of the scale using corner points [10]. Ledwich and Williams [11] reduced the complexity of SIFT features and the number of features that define the environment by working indoors. The matching time was shortened with reducing the size and complexity of the features. Guan et al. [12] developed an algorithm called Speeded-Up Robust Features (SURF) to extract local features and so they performed object recognition quickly and efficiently. The studies showed that this algorithm is invariant to rotation and scale of the image as well as robust to illumination changes. Besides, they concluded that this algorithm is more robust than the cases where only the SIFT and Principal Component Analysis (PCA)-SIFT algorithms are used together. Heo et al. [13] extracted the features using

978-1-5386-6878-8/18/$31.00 ©2018 IEEE

Features from Accelerated Segment Test (FAST) and Binary Robust Independent Elementary Features (BRIEF) named algorithms together. They concluded that these algorithms are at least fifty-fifty better in terms of speed and memory capacity than any of SIFT or SURF algorithms. Rublee et al. [14] used an algorithm called Oriented FAST and Rotated BRIEF (ORB), which aims to combine the good properties of BRIEF and FAST algorithms as an alternative to these algorithms because SIFT and SURF use a large number of features for object detection and matching.

In this study, using 4 degrees of freedom (DOF) OWI–535 robotic arm, it is purposed that the recognized object is grasped in the workspace of the robotic arm and dropped to the desired target. For this purpose, an appropriate experimental setup is established. Object recognition and localization operations are performed with software written in C++ language using Open Source Computer Vision (OpenCV) library on Raspberry Pi, which is a Single-Board Computer (SBC) based on Linux operating system. With this software, firstly, images of desired objects are taken by camera and database is created. Secondly, by applying local feature-based object detection and recognition algorithms named SIFT, SURF, FAST, and ORB to the taken test images, it is determined which object is registered in the database. Thirdly, the center point of the recognized object is determined in terms of pixels and converted into position information that the gripper must reach. Finally, this position information is sent via serial communication from Raspberry Pi to Arduino Mega board based on the microcontroller which will carry out motion of the robotic arm. Such a study is performed for the first time on OWI–535 robotic arm. This study shows that local feature-based algorithms are useful for applications of education and industry.

The rest of this paper is organized as follows. In Section 2, first, the necessary steps for object recognition and localization are given. Then, object detection and recognition algorithms used in feature extraction are given in detail. In Section 3, the parts of the experimental setup are mentioned. In Section 4, the comparative results of the applied local feature-based algorithms on various test images are given. In Section 5, the paper is concluded.

II. OBJECT RECOGNITION AND LOCALIZATION

In recent years, in areas where the error is not acceptable, in order to implement applications such as object tracking, image matching, object detection and object recognition are preferred local feature-based algorithms for extracting the features. Because, these algorithms are more robust to external factors such as illumination changes, occlusions, scale changes etc. The features extracted using local feature-based algorithms are expressed as keypoint descriptors in the image. These keypoints (interest points) in the image can be patch, edge, corner or blob, and simplify object recognition by eliminating unnecessary information from the image.

In this study, local feature-based algorithms are used for feature extraction. The features are extracted by detecting the keypoints in the object images and calculating the descriptors of these keypoints. The general stages of the implemented local feature-based object recognition and localization system are given in Fig. 1.

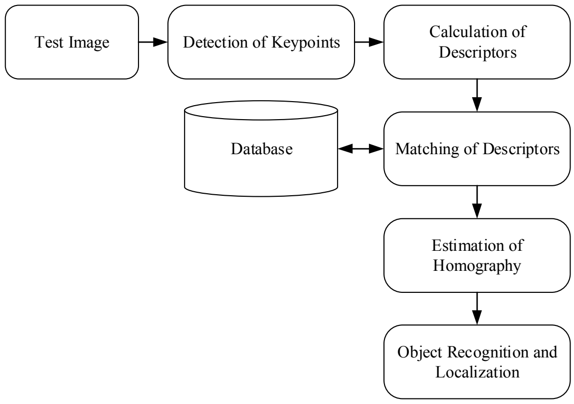


Fig. 1. The overall workflow diagram of local feature-based object recognition and localization

A. Feature Extraction

An object image to be classified in the object recognition process usually contains a lot of unnecessary information. This reduces the precision of the classification and increases the processing time. To avoid this negativity, the object information is transformed into another database at a lower size. The feature extraction is a transformation process in which the excess and unnecessary data of the object are eliminated.

In this study, it is the main objective to recognize and localize the object. Suitable features affect positively the success of applications such as object detection and recognition. For this reason, SIFT, SURF, FAST, and ORB algorithms which have proven successful in the scientific literature are used to extract features. The processing steps of the local feature-based algorithms used in this study are performed after converting the images from RGB to grayscale.

In 2004, the algorithm shortly called SIFT was proposed by Lowe to extract features in the image. SIFT algorithm can extract features without being affected by illumination conditions, small angular changes in the image and scale of the image [15]. On the other hand, although the algorithm is suitable for detecting objects with high-resolution images, the algorithm is slow in terms of execution time [16].

The corner is the most efficient one as a keypoint, and for this reason, the corners are detected and others are eliminated. The "Hessian matrix" is used to determine whether a point is a corner. After keypoints are detected and gradient orientations are determined, features are obtained robust to scale, rotation and position. After these operations, keypoint descriptors representing the feature vectors are obtained as shown in Fig. 2.

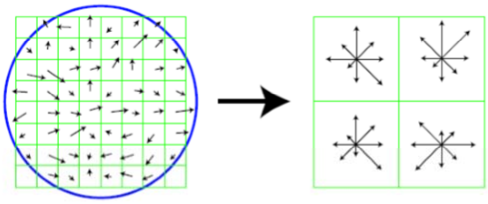


Image gradients Keypoint descriptor

Fig. 2. Obtaining image gradients and keypoint descriptor [15]

In 2008, in order to extract features in the image, Bay proposed an algorithm shortly called SURF, which is based on the calculation of the Hessian matrix. Since integral images are used in the Hessian calculation, the calculation time is reduced, which allows the algorithm to run faster than SIFT algorithm [17].

The box filters are obtained using an integral image. The keypoints are detected by means of the scales obtained from the box filters. After the keypoints are obtained, a circular region is selected around the keypoints, and Haar wavelet filters are applied to this region to extract the descriptors for each keypoint [18]. As shown in Fig. 3, components obtained from the Haar wavelet filter are found for each region in the 4×4 region [17].

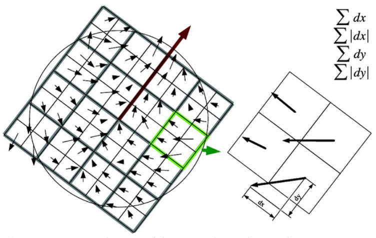


Fig. 3. Creation of keypoint descriptors [17]

In 2006, Rosten and Drummond proposed an algorithm shortly called FAST in order to detect keypoints in the image [19]. FAST performs the detection of the keypoints by detecting corner points and has high speed and reliability [16].

In order to be a fast and effective alternative to the SIFT or SURF algorithms in feature extraction of the image, an algorithm shortly called ORB was proposed by Rublee et al. in 2011 [14]. ORB algorithm uses FAST for detection of keypoints and BRIEF developed in recent years for calculation of descriptors. The features extracted by this algorithm are both insensitive to rotation and noise and robust to illumination changes.

Figure 4 shows the representation of the keypoints detected by SIFT, SURF, FAST, and ORB algorithms of an object image used in the study.

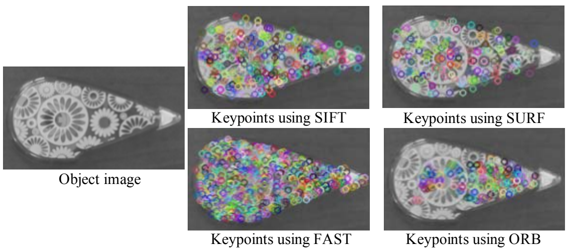


Fig. 4. Sample object image and detection of its keypoints with different algorithms

B. Feature Matching and Classification

In order to classify an unknown object in a given test image after the database is created by extracting the features of the training images, the features that are initially included in the object database are individually matched with each of the features extracted from the test image. The matches give information about the object class.

The similarity measure between descriptors depends on the data type. Hamming distance is preferred for binary data types. Euclidean distance is available for real-valued data. Euclidean metric is usually used to calculate distances between the descriptors.

Fast Library for Approximate Nearest Neighbors (FLANN) [20] algorithm, proposed by Muja and Lowe and written in C++, is based on the nearest neighbor search and consists of collection algorithms such as hierarchical k-means tree [21] and multiple randomized kd-trees [22]. After the descriptors representing the features are matched and the distances between them are calculated, the closest matches are determined.

Another matching algorithm, Brute-Force (BF), is used with any of the Euclidean or Hamming metrics. In this algorithm, the closest descriptors are found by performing all combinations of matches between the database and the descriptors obtained from the test image.

C. Homography Estimation

Homography is a two-dimensional projection transformation that matches the points on a plane to the other [23]. Some mismatches can occur when keypoint descriptors extracted from images are matched to obtain similar points. In this study, the algorithm called Random Sample Consensus (RANSAC) [24] was used because it can detect the most accurate matches by eliminating mismatches between these similar point set.

The RANSAC algorithm proposed by Fischler and Bolles is a general parameter estimation approach designed to overcome a large number of outliers in the input data [25]. By using the RANSAC algorithm, the homography estimation is performed. As a result of obtaining the most accurate matched descriptors, the position of the object is correctly determined. Fig. 5 shows the matches between a sample object image and a test image. As it can be seen, the RANSAC algorithm provides that the appropriate point pairs are enclosed with a bounding box by eliminating outliers.

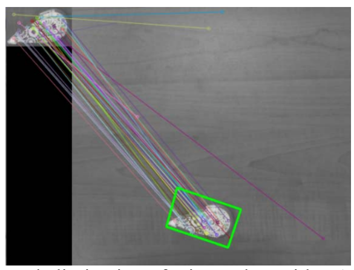


Fig. 5. Detection and elimination of mismatches with RANSAC algorithm

III. EXPERIMENTAL SETUP

It is aimed respectively to orient a recognized object or objects in the workspace of the 4 DOF OWI–535 robotic arm, grasp it and drop it to a destination using local feature-based object detection and recognition algorithms. The experimental setup established for this purpose is shown in Fig. 6.

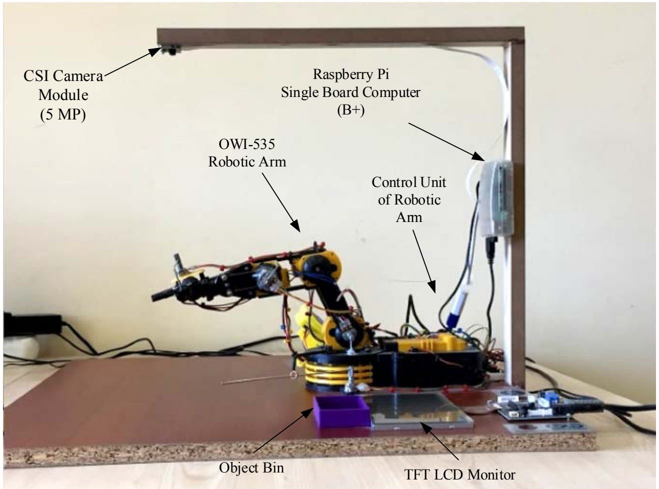


Fig. 6. Experimental setup

The experimental setup consists of two main parts. The first of these is Raspberry Pi SBC on which object recognition and localization processes are performed and the hardware and software requirements for the processes to be performed using this SBC. In the study, B+ model of Raspberry Pi was used. This model is based on the Broadcom BCM2835 System on Chip (SoC), which includes a single-core ARM1176JZ-F 700 MHz processor unit with ARMv6 architecture. The second is OWI–535 robotic arm and its control unit, which provides the recognized and located object to be grasped and dropped to the object bin. The joint angles that need to be known for the gripper to reach the desired position are obtained by inverse kinematic analysis of OWI–535 robotic arm. In this study, for the inverse kinematic analysis of the robotic arm, the equations obtained in [26] were used.

Recognition of the object in the image captured by the camera and detection of the location were performed with software developed using the OpenCV library in C++ language on Raspberry Pi.

OWI–535 robotic arm has been released wired control. For our study, the wires of the motors in the robotic arm joints were removed, and a control unit independent of the wired control was built up. Since the joints are designed as DC motor-gearbox pair, it is almost impossible to replace the motors in the joints with servo motor which can receive position information by feedback. Therefore, potentiometers are placed on each joint of the robotic arm and the gripper to provide feedback as shown in Fig. 7. The potentiometer, which is placed for the gripper, enables the control of whether gripper is open or closed. By correlating the information read from the potentiometers and the joint angle, a feedback system is obtained and position control of the joints is carried out. Besides, Force Sensitive Resistor (FSR) is placed on one side of the gripper to determine how much the gripper should be closed during the grasping of the object.

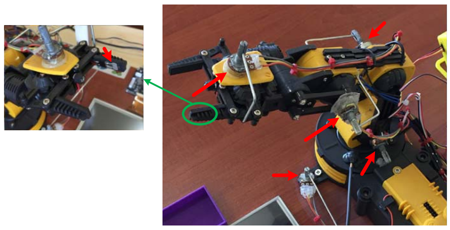


Fig. 7. Placement of FSR and potentiometers

For the control of OWI–535 robotic arm, firstly, the cables of the motors, which are connected to the wired control, were disassembled. Then, the control unit was set up to provide control of the motors.

IV. EXPERIMENTAL RESULTS

In experimental studies, the performances of SIFT and SURF algorithms, which are best known in the scientific literature, were compared and then object recognition and localization processes were performed to grasp the object with the gripper of the robotic arm. The process steps given in Fig. 8 were carried out for the experimental studies.

SIFT, SURF, and ORB algorithms were used alone because they can both detect keypoints and calculate descriptors of keypoints. However, since FAST algorithm does not have its own keypoint descriptor, the descriptors were calculated using the SURF algorithm in experimental studies. Parameters of the algorithms in OpenCV library were adjusted in accordance with the study in the detection of keypoints. The number of images within each octave of a Gaussian pyramid was set to 3 in SIFT, 2 in SURF. The contrast threshold used to filter out weak features in semi-uniform regions was set to 0.04 in SIFT. The threshold for Hessian keypoint detector was set to 200 in SURF. The threshold on the difference between the intensity of the central pixel and pixels of a circle around this pixel was set to 20 in FAST. The maximum number of features was set to 500 in ORB.

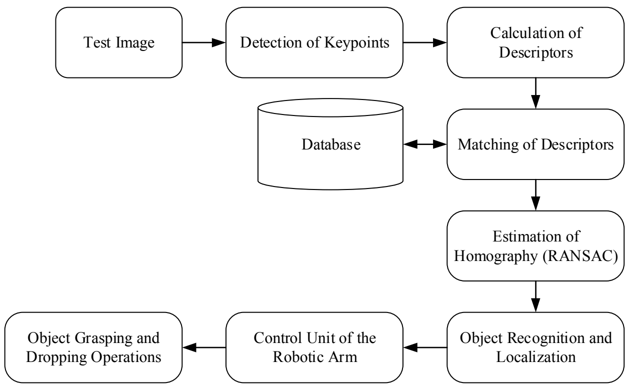


Fig. 8. Process steps for experimental studies

For the experimental studies, the database contains the feature vectors of the object images in small size because the objects used in training phase of the system are the objects that can be grasped by the gripper. Furthermore, since the used camera is two-dimensional, the height of the objects was selected very close to each other. Thus, the gripper can perform the grasping operation at the fixed z-position coordinate (p_z). After images of objects randomly placed in accordance with a workspace of the robotic arm were taken by the camera as RGB at 1024×768 resolution, only images belonging to the objects were cropped. As shown in Fig. 9, the database was created with 10 objects. In the training phase of the object recognition and localization studies, 5 different object images were used for each class to include difficult conditions such as illumination changes, rotations or scale differences due to camera viewpoint.

Fig. 9. RGB images of the objects used for database

The labels of the object classes in Fig. 9 are given respectively; *Blue Eraser with Character*, *Red Eraser with Character*, *Eraser with Note Symbol*, *Eraser with Piano Symbol*, *Pink Correction Fluid*, *Blue Correction Fluid*, *Green Correction Fluid*, *White Eraser with Character*, *Sarajevo Memory*, *Ultrasonic Sensor*. The descriptors were stored together with the class labels to obtain the feature database required for the training phase. Keypoint detector and keypoint descriptor algorithms were applied to the training images as SIFT+SIFT, SURF+SURF, FAST+SURF, and ORB+ORB. The numbers of the total feature vectors of the databases obtained by these algorithms are given in Fig. 10.

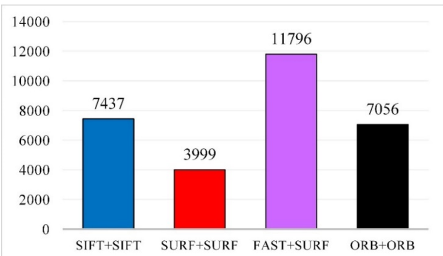


Fig. 10. Total numbers of feature vectors extracted by different algorithms

Test images used in experimental studies are scene images that contain both objects used in the training of the system and different objects. A total of 20 test images were taken, including 2 test image for each object class. These test images, taken as RGB at 1024×768 resolution, were converted to grayscale. Thus, they were made ready for the test phase.

After the test images were identified, the descriptors obtained by SIFT and SURF were matched using FLANN-based matcher, and the descriptors obtained by ORB were matched using BF matcher. The Hamming metric was used to match binary descriptors calculated by ORB.

After the test images were identified, the descriptors obtained by SIFT and SURF were matched using FLANN-based matcher.

The experimental results obtained by using SIFT+SIFT and SURF+SURF algorithms are given in Table I.

TABLE I. THE EXPERIMENTAL RESULTS (SIFT+SIFT AND SURF+SURF)

Test image number	SIFT+SIFT		SURF+SURF	
	Number of matched descriptor pairs	Number of outliers	Number of matched descriptor pairs	Number of outliers
1	95	0	129	6
2	104	3	45	3
3	52	0	50	9
4	69	0	32	1
5	144	0	134	134
6	82	0	88	12
7	25	0	31	0
8	39	0	25	0
9	19	0	12	0
10	35	0	22	0
11	6	0	46	4
12	36	36	51	1
13	69	0	24	0
14	53	0	18	0
15	24	0	32	0
16	37	0	32	1
17	36	0	38	0
18	32	0	25	0
19	8	0	13	0
20	11	0	9	0

The percentage of correct matches of the descriptors obtained by the SIFT+SIFT and SURF+SURF algorithms according to Table I is about between 97%-100% and 82%-100%, in other images, except for completely wrong matches of test image 12 and 5, respectively. The sample test operations are shown in Fig. 11 and Fig. 12.

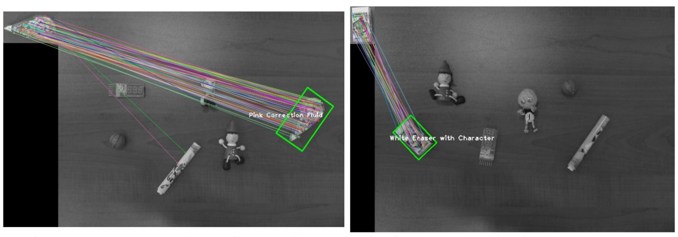


Fig. 11. Test operations for test image 2 and 10, respectively (SIFT+SIFT)

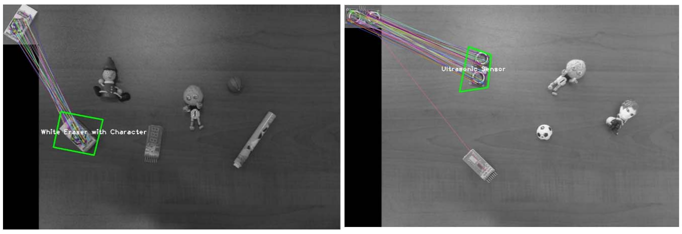


Fig. 12. Test operations for test image 10 and 12, respectively (SURF+SURF)

The experimental results obtained by using FAST+SURF and ORB+ORB algorithms are given in Table II.

TABLE II. THE EXPERIMENTAL RESULTS (FAST+SURF AND ORB+ORB)

Test image number	FAST+SURF		ORB+ORB	
	Number of matched descriptor pairs	Number of outliers	Number of matched descriptor pairs	Number of outliers
1	233	32	45	2
2	145	12	35	2
3	140	14	134	42
4	90	11	5	0
5	162	10	178	178
6	152	7	46	4
7	53	12	6	0
8	26	2	7	0
9	16	0	7	0
10	31	1	19	0
11	152	152	313	313
12	69	7	313	313
13	49	2	5	0
14	45	0	10	0
15	66	1	20	0
16	34	0	27	0
17	31	0	8	0
18	14	0	13	0
19	23	0	16	0
20	40	0	8	0

The percentage of correct matches of the descriptors obtained by the FAST+SURF and ORB+ORB algorithms according to Table II is about between 77%-100% and 68%-100%, in other images, except for completely wrong matches of test image 11 and 5, 11, and 12, respectively.

The comparison of the numbers of the keypoints in the test images is shown in Fig. 13. In addition, the comparison of the feature extraction times, which represent the total time elapsed to detect the keypoints in the test images and to calculate the descriptors, is also shown in Fig. 14.

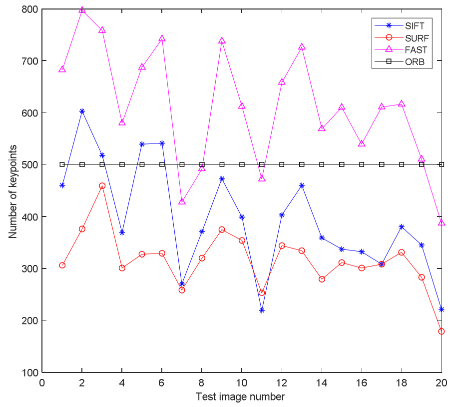


Fig. 13. Comparison of the numbers of the detected keypoints

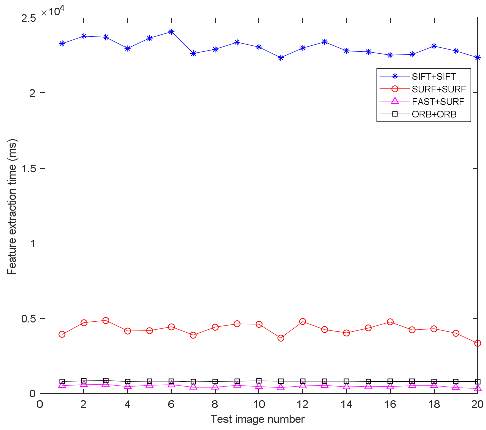


Fig. 14. Comparison of feature extraction times

In order to compare the execution times of the feature extraction algorithms used, the feature extraction times per keypoint were considered. For this, in Table III, the averages of numbers of the keypoints in Fig. 13 and the feature extraction times in Fig. 14 obtained for each test image are taken.

TABLE III. COMPARISON OF THE EXECUTION TIMES OF THE ALGORITHMS

Algorithm	Average number of keypoints	Average feature extraction time (ms)	Time/Number of keypoints
SIFT+SIFT	395.35	23,048	58.2977
SURF+SURF	316.4	4,274.7	13.5104
FAST+SURF	610.7	481.47	0.7884
ORB+ORB	500	808.865	1.6177

According to Table III, it is observed that FAST+SURF is the fastest algorithm. Besides, ORB+ORB may be a good alternative to FAST+SURF in terms of speed.

After object detection and recognition are performed, the x and y position coordinates, which must be reached for the grasping of the object, are calculated by converting pixel-mm thanks to the information in Fig. 15.

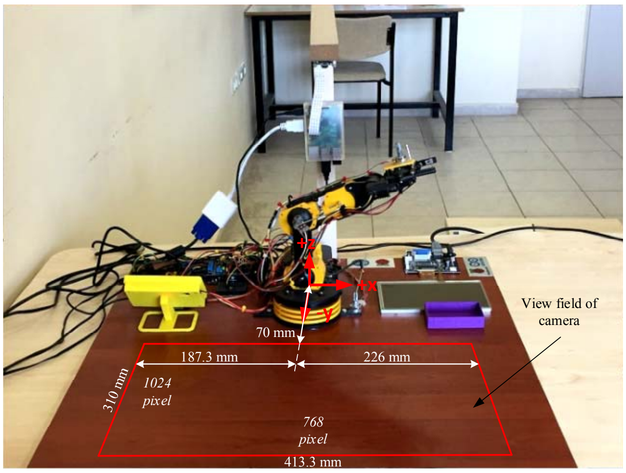
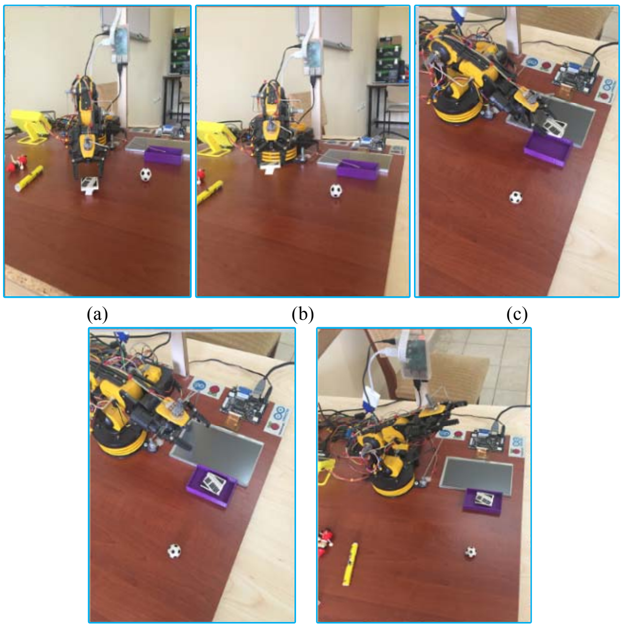


Fig. 15. Necessary information for pixel-mm conversion

p_z was determined a constant value because the heights of the objects were selected very close to each other as mentioned before. In our study, this value was set as 30 mm and found suitable. In addition, since the robotic arm has no 5th axis on its gripper, the object was placed in such a way that the gripper can grasp it.

Experimentally visualize the processes carried out to recognize and move the object. The coordinates of the center point of the recognized object were determined in terms of the pixel, and the position coordinates needed to reach the gripper with the necessary calculations were obtained. After the gripper reaches the specified position, the motions performed are as shown in Fig. 16, respectively.



(a) (b) (c) (d) (e)

Fig. 16. Motions order of the robotic arm: (a) reaching to the object, (b) grasping the object, (c) moving the object, (d) dropping the object, and (e) moving to the home position

V. CONCLUSIONS

In this study, an experimental setup was established to recognize an object, to determine the position of the object, and to carry out the process of grasping and moving the object with the aid of the gripper of the robotic arm. Object recognition and localization processes were accomplished with software written in C++ using the OpenCV library and embedded in Raspberry Pi SBC.

Different images were taken for both the training phase and the test phase in order to perform object recognition and localization processes experimentally. The performances of the local feature-based algorithms used on these images were compared.

SIFT+SIFT, SURF+SURF, FAST+SURF, and ORB+ORB keypoint detector and keypoint descriptor algorithms were used for experimental studies. The results of object recognition with these algorithms have proven to be quite successful. In the feature matching operation performed for the recognition process, the most accurate matches have been successfully achieved with the SIFT+SIFT algorithm about between 97% and 100%. The SIFT+SIFT algorithm was followed by SURF+SURF in terms of matching performance. It has once again been shown that the FLANN-based matcher for SIFT and SURF descriptors and the BF matcher for ORB descriptors are suitable for use with the Hamming metric as described in the literature. In order to compare the execution times of the algorithms with each other, the feature extraction times per keypoint was taken as reference. FAST+SURF and ORB+ORB were obviously the fastest algorithms. The SIFT+SIFT algorithm, on the other hand, appeared to work much slower than other algorithms, in contrast to the best matching performance.

REFERENCES

[1] Y. Mingqiang, K. Kidiyo, and R. Joseph, “A survey of shape feature extraction techniques”, *Pattern Recognition*, IN-TECH, pp. 43–90, 2008.K. Elissa, “Title of paper if known,” unpublished.

[2] H. Morevec,“Rover visual obstacle avoidance”, *Proceeding of the 7th International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 2, pp. 785–790, 1981.

[3] C. Harris and M. Stephens, “A combined corner and edge detector”, *4th Alvey Vision Conference (AVC)*, vol. 15, pp. 147–151, 1988.

[4] C. Harris, “Geometry from visual motion”, *Active vision*, MIT Press, pp. 263–284, 1993.

[5] C. Schmid and R. Mohr, “Local grayvalue invariants for image retrieval”, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 19, no. 5, pp. 530–534, 1997.

[6] K. N. Shree, “Visual learning and recognition of 3-D objects from appearance”, *International Journal of Computer Vision (IJCV)*, vol. 14, no. 1, pp. 5–24, 1995.

[7] M. Swain and D. Ballard, “Color indexing”, *International Journal of Computer Vision (IJCV)*, vol. 7, no. 1, pp.11–32, 1995.

[8] B. Schiele and J. L. Crowley, “Object recognition using multidimensional receptive field histograms”, *4th European*

Conference on Computer Vision (ECCV), vol. 1, pp. 610–619, 1996.

[9] C. Shanthi and N. Pappa, “An artificial intelligence based improved classification of two-phase flow patterns with feature extracted from acquired images”, *ISA Transactions*, vol. 68, pp. 425–432, 2017.

[10] D. G. Lowe, “Object recognition from local scale-invariant features”, *IEEE 7th International Conference on Computer Vision (ICCV)*, vol. 2, pp. 1150–1157, 1999.

[11] L. Ledwich and S. Williams, “Reduced SIFT features for image retrieval and indoor localisation”, *Australian Conference on Robotics and Automation (ACRA)*, vol. 322, p. 3, 2004.

[12] F. Guan, X. Liu, W. Feng, and H. Mo, “Multi target recognition based on SURF algorithm”, *6th International Congress on Image and Signal Processing (CISP)*, vol. 1, pp. 444–453, 2013.

[13] H. Heo, J. Y. Lee, K. Y. Lee, and C. H. Lee, “FPGA based Implementation of FAST and BRIEF algorithm for object recognition”, *TENCON 2013-2013 IEEE Region 10 Conference (31194)*, pp. 1–4., 2013

[14] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: an efficient alternative to SIFT or SURF”, *International Conference on Computer Vision (ICCV)*, pp. 2564–2571, 2011.

[15] D. G. Lowe, “Distinctive image features from scale-invariant keypoints”, *International Journal of Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91–110, 2004.

[16] H. Joshi and M. K. Sinha, “A Survey on image mosaicing techniques”, *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 2, no. 2, pp. 365–369, 2013.

[17] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (SURF)”, *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[18] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors”, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 27, no. 10, pp.1615–1630, 2005.

[19] E. Rosten and T. Drummond, “Machine learning for high speed corner detection”, *Proceeding of the 9th European Conference on Computer Vision (ECCV)*, vol. 1, pp. 430–443, 2006.

[20] M. Muja and D. G. Lowe, “Fast approximate nearest neighbors with automatic algorithm configuration”, *International Conference on Computer Vision Theory and Applications (VISAPP)*, vol. 1, pp. 331–340, 2009.

[21] L. Y. Pratt, “Comparing biases for minimal network construction with back-propagation”, *Part of: Advances in Neural Information Processing Systems*, Morgan Kaufmann Publishers Inc., pp. 177–185, 1989.

[22] C. Silpa-Anan and R. Hartley, “Optimised KD-trees for fast image descriptor matching”, *IEEE Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8, 2008.

[23] D. Vaghela and P. Naina, “A review of image mosaicing techniques”, *International Journal of Advance Research in Computer Science and Management Studies (IJARCSMS)*, vol. 2, no. 3, pp. 431–437, 2014.

[24] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”, *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[25] K. G. Derpanis, “Overview of the RANSAC algorithm”, *Image Rochester NY*, vol. 4, no. 1, pp. 2–3, 2010.

[26] C. Kaymak and A. Ucar, “Kinematic Analysis of OWI–535 Robotic Arm and Simulation of Its Motion using SimMechanics”, *8th International Advanced Technologies Symposium (IATS)*, pp. 2468–2476, 2017.