Fusion of Camera-Acquired Data and CAD 3D Models of Objects in Forming a Visual Feedback Loop for Industrial Robots

Miloš Nenadović, Uroš Ilić, and Aleksandar Rodić

***Abstract*—In this paper, we present a computer vision system for object detection and spatial position and orientation recognition. To solve the problem, we separated the process in several stages: in the first stage the system uses principal component analysis (PCA) method in optimal conditions to detect objects in the image. Then, after the object is extracted, we projected it in the appropriate eigenspace, produced by singular value decomposition (SVD) of the set of images of the rotated 3D CAD model. The closest match is then processed by correlation between it and the real-object image in log-polar space. The result is combined with information from other cameras to derive the approximate position and object orientation using multiple-view geometry.**

***Key words*—Object detection; Singular Value Decomposition; Log-Polar; Multiple-view geometry.**

I. Introduction

The problem of object detection in images is one of the oldest in computer vision [1] and has over the years been solved by various methods, among which are neural networks and principal component analysis (PCA). Convenient in the sense that they only require a set of training data to extract key features from objects, they provide vastly accurate classification results. However, as the accuracy rises so does the required memory, which is also the case as the complexity of objects rises.

In industrial robotics we require the robot arm to handle objects of varying complexities, ranging from simple boxes to machine parts. In these cases, a camera is a good tool to use to scan the scene and retrieve information about objects. The problem arises when we need to do more than classify an object. Assuming we are only provided with several cameras, we aim to get the most information about the scene which would be the position and orientation of objects.

We will aim to decrease the amount of images necessary to train the neural network, or in the case of principal component analysis [2] to eliminate the need for them altogether and instead only use the 3D CAD models provided. The CAD models are used to obtain reference images of their respective objects in different positions [3]. From these images we calculate the histogram of oriented gradients (HOG) and use them for singular value decomposition (SVD) based on the idea that these features are enough to determine, at least the approximate, orientation of an object. Following this, we will use correlation of gradient data in log-polar space which is rotation and scale invariant. Additionally, we will analyze the speed of the algorithm to determine whether it can be used for moving objects. We will test the system on a set of simple objects (cube, cuboid, cylinder and pyramid) of different colors (red, blue, green, brown, orange, black). In future work, we will attempt to use the system on more complex objects such as machine parts or 3D wooden puzzle pieces.

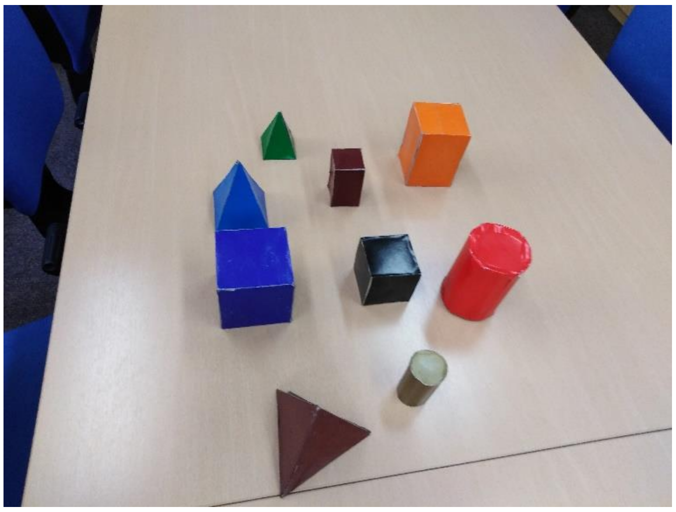
Fig. 1. Scene setup with simple objects of multiple colors on a table.

II. The Role of Robot Vision and 3D CAD Modeling in Trajectory Planning

To understand the need for proper object position and orientation recognition, we look at an example of a robot grappler arm. We assume the objects are still and placed on a table in various positions. To successfully grab an object, the robot arm must be placed so that the object does not slip from its grasp. Additionally, a robot may need to navigate between objects and in such cases the knowledge of their position and

Miloš Nenadović is with the Faculty of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: neosmilos@gmail.com).

Uroš Ilić is with the Faculty of Mechanical Engineering, University of Belgrade, 16 Kraljice Marije, 11120 Belgrade, Serbia (e-mail: ilicuros01@gmail.com).

Aleksandar Rodić is with the Institute Mihajlo Pupin, 7 Sazonova, 11000 Belgrade, Serbia (e-mail: aleksandar.rodic@pupin.rs).

orientation is crucial [1]. Camera images can be used to determine the approximate position of an object using multiple-view geometry, after knowing the exact or approximate positions of the cameras. In this case we assume camera positions are unknown except for their general relative position i.e. ‘front’, ‘left’ and ‘above’. Thus, we used reference points to determine the approximate camera positions before using them to determine the object positions.

III. Configuring the 3D Robot Vision System

Although some results are possible without calculating the positions of the cameras, we strive to achieve the most accurate representation of the scene and so require this as the first step. Given that no prior information is known about the position and angle at which the cameras observe the scene, we use a set of reference points on the scene to determine them. The size, position and distance between the reference points is known. We search the images for these points, assign the appropriate labels for them, and then, using the known data we reconstruct the camera position and orientation.

IV. Fusion of 3D CAD Model Data and Camera Images

The CAD models are used to construct a set of images from various views by applying rotation transformations to them. For the SVD algorithm we use rotations around each of the axes in increments of 15 degrees, whereas for the log-polar correlation we use 5-degree increments, however for only two axes. This is because one of the axes contributes solely to 2D image rotation.

As for the vision data, as mentioned before, we use a set of three cameras to observe the scene. Each camera image is processed on its own in parallel with the others and upon completion updates the other two. In this way, should any image be wrongly paired, the image with the higher correlation coefficient will take precedence and change the initially estimated view-image to another one based on the camera positions.

In this way we achieve a feedback loop by utilizing CAD model data to interpret the information acquired by cameras and then comparing them to the real thing.

V. Recognition of Object Shape and Spatial Orientation using Images and 3D CAD Models

A. *Object detection using PCA*

Before the object orientation algorithm can begin, we must first detect objects in the image. One of the goals we set ourselves was to detect the objects without using real-object images, but rather just the models. For this purpose, we created 936 training images for each of the models, depicting different views that we used as the training set.

The PCA method for object detection relies on using a training set to find principal components (PCs) to use to reconstruct images. Assuming that the best image reconstruction can be done only when using the PCs created from an image set of the same type [2], based on the accuracy of the reconstruction, we can classify an image as one of the objects or part of the background.

As mentioned in [2] we use vectorized reference intensity images to acquire a projection matrix P, and μ as the mean of the set. When given a vectorized intensity input image u we get the reconstructed image r through:

$$r = P'P(u - \mu) + \mu. \tag{1}$$

The reconstructed image is then compared to the original image and their difference d is expressed as the 2-norm between the two image vectors, as in

$$d = \|r - u\|. \tag{2}$$

To test this method, we supplied an image of the blue pyramid object as the input image and then used different training sets containing reference model-images and looked at differences d they produced.

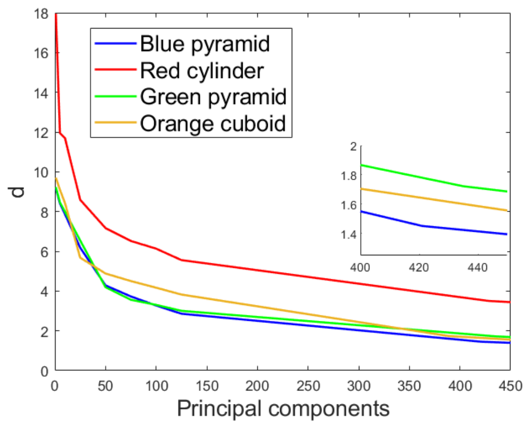

Fig. 2. Image reconstruction of the blue pyramid image using principal components obtained from four different model data sets (blue pyramid, red cylinder, green pyramid, and orange cuboid).

As shown in Fig. 2, the best reconstruction is achieved by using the appropriate training data set. We used a set of fixed numbers for the number of PCs used to reconstruct the image, except for the final one, which was the number of PCs that conserved 90% of the singular energy α, similar to how it was done in [3].

One of the problems this method encountered was the situation when, instead of an object, the input image was that of the background. Given its simplicity in comparison to an image containing an object the difference d was even lower than those of the desired object. Thus, we found that it was necessary to have a minimum number of real-object images and background images to create an additional classifier, or for the parts of the image with just the objects to be extracted before applying PCA.

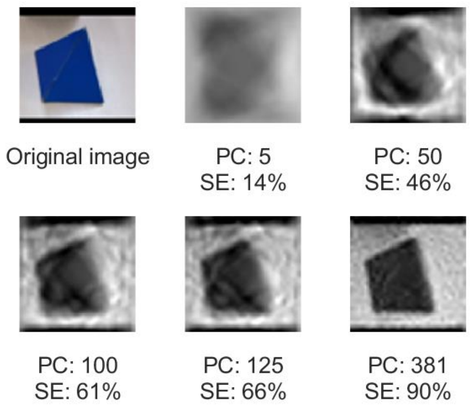

Fig. 3. Image reconstruction using principal components obtained from the blue pyramid model data set, with PC showing the number of principal components used and SE representing the retained percentage of singular energy.

As shown in Fig. 3 increasing the number of PCs also increases the accuracy of the reconstruction. However, doing so beyond a certain point would reduce efficiency instead as the increase of accuracy does not justify the decrease in computation speed and added memory.

B. Finding the closest match using the SVD algorithm

After extracting the object from the entire image, we proceed to use singular value decomposition to find the closest match among a set of rotated model images. As an object's orientation can be represented by its edges, we use gradient analysis to find features. The histogram of oriented gradients provides us with the necessary data t (column matrix of size $n \times 1$) that we place as the columns of a matrix $T = [t_1\ t_2\ \dots\ t_N]$, which the SVD algorithm is applied to.

$$T = U\Sigma V^T \tag{3}$$

From (3) we take the matrix U and use the first k ($1 \leq k \leq N$) columns representing eigenvectors of the largest k eigenvalues in S, sorted in descending order [2]. Like so, we derive the matrix U' from U. Multiplying with vectorized HOG features of input image data u we can project that data to the eigenspace, while reducing dimensionality. This is also applied to all columns of the matrix T creating a matrix T' of reduced size $k \times N$. We then compare the projected data of the input image u with the other projections t' from reference image HOG features (columns of T'), as in

$$d_i = \left\| t_i - U^{T'} u \right\|. \tag{4}$$

Index i in (4) denotes the index of the projection being compared with the projected input image. We store the results in an array which we sort at the end of the algorithm, and the index with the minimum difference points to the reference model image that would be the best pair to the input image.

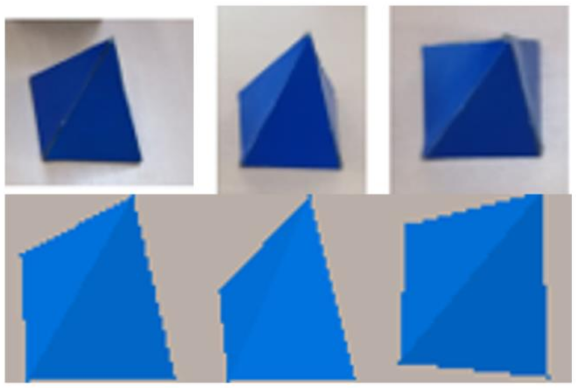

Fig. 4. Result of SVD comparison between extracted object images (top row) from three views (front, left and above) and a set of rotated model images (bottom row).

We can see in Fig. 4 that the algorithm provides two good matches and one bad match. Additionally, we can see that the sides of the pyramid in bad lighting were omitted from the approximations. However, as the algorithm's main purpose is just to narrow the search range for the subsequent parts, having even two out of three bad matches wouldn't pose a problem.

In future work, we will focus on improving the quality of the SVD method to include more accurate estimates both in optimal conditions, as was done in this paper, and when there is occlusion as the total time required for the match to be found greatly depends on the initial estimate.

C. Finding the closest match using log-polar correlation

Aside from the SVD algorithm used to narrow down the possible choices for object positions, we use intensity image correlation in log-polar space to deduce the best pairing of the real-object-image and view-image of the model. Log-polar space is used as it is both rotation and scale invariant [5]. Correlation is used to find the pair with the highest correlation coefficient and determine the rotation angle which is proportional to the vertical shift.

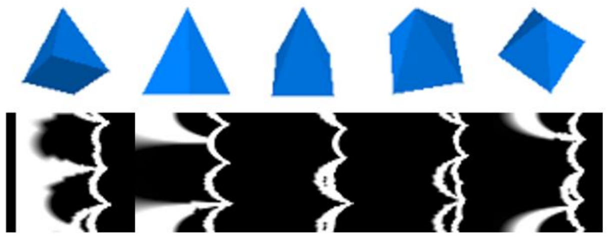

Fig. 5. Model images from multiple viewpoints (first row) and the log-polar transformations of the paired intensity images (second row)

One thing that can be seen from Fig. 5 is that all the log-polar transformations are quite unique. As such, it is not required to store all the information from those images. Rather, as they are computed from the center of the object and images encompasses mainly the object in question, we can take an area of interest that is the second half of the log-polar image along the ρ-axis. This area contains the entire object outline as well as the endpoints of the object edges within the outline.

Problems that may occur in this stage are related to the center used for computing the log-polar transformation. Because we were dealing with objects without occlusion and that are distinct from the background, we could use simple methods of color segmentation to find all the object pixels and then determine the center as a "center of mass".

However, in cases where occlusion is present, or the object is more complex, or the object detection algorithm does not perfectly capture the object, finding the exact center becomes a problem. As mentioned before, log-polar correlation is both rotation and scale invariant, but it is sensitive when it comes to the center from where it is calculated. In case the estimated center of the object is shifted from where it is located on a template image then correlation may not yield adequate results, depending on how much the center is shifted. In [5] it is shown that the best correlation coefficient is achieved when the location of the object center matches that of its template image.

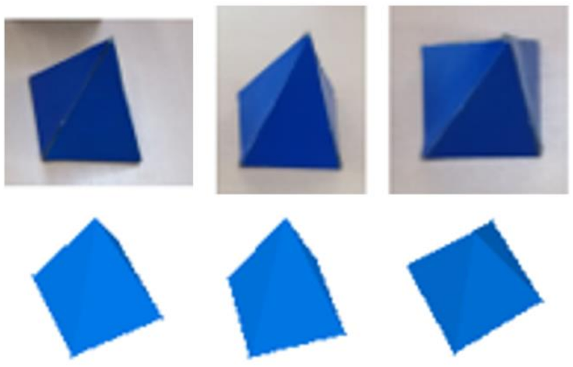
Fig. 6. Real-object images (first row) and best pairs from log-polar correlation without angle correction (second row).

The results shown in Fig. 6 show that the log-polar correlation method finds accurate matches for the objects in question. This means that it is safe to use the less-accurate SVD method first to provide an initial object orientation estimate.

VI. Experimental Results and Discussion

The algorithm was run under the assumption of real-time work, meaning that the number of additional angles around the ones found by the SVD method, which would be checked by the log-polar correlation, had to be lowered to two sets of 4 angles on each side of the central one found by SVD (total of 9 angles). With such modifications the algorithm ran at around 60 ms for single-object orientation detection. This is excluding the time for object detection which took around 50 ms. We will attempt to further bring down the required computation time for this task in future work.

Additionally, one of the possible alterations that we tried was lowering the number of angles that would be checked to only one, the one provided by the SVD method. This lowered the computation time from 60 to around 10 ms.

The possibility of larger errors occurring due to SVD was taken into account and thus after every iteration (determining the 6D position of every object in the scene), the results are carried over to the next iteration until a plateau is hit. Based on the experimental results, we have determined that the algorithm is capable of operating in real-time.

The entire algorithm we proposed is an iterative one and can be summarized in several steps:

1. We use PCA to detect objects in the image for each of the cameras. This step can be skipped after the first run in cases where the scene is static.
2. We calculate an input image's HOG features and send them to the SVD algorithm to estimate the object's orientation. This is done for every camera image, resulting in three estimates.
3. The estimates are then turned to intensity images, transformed into log-polar space, and processed by 2D correlation. The resulting coefficients are compared, and the best result is sent further.
4. Knowing the transformation matrices between cameras, we use the estimate to obtain images that the other cameras should see. These images are treated as the new input images and sent to Step 1.

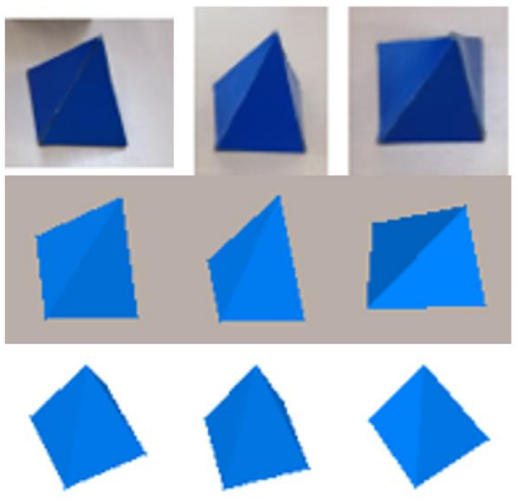
Fig. 7. Iteration 1 of the algorithm. Real-object images (first row), SVD estimated object view images (second row) and log-polar correlation results without angle correction (third row).

Fig. 7 shows that even in the first step of the algorithm we can obtain accurate results, similar to ones in Fig 6 and that the algorithm would end in the second iteration. The total number of iterations needed for convergence varies depending on the initial orientation estimations as well as the number of camera images that we can work with. In cases where one or two of the cameras cannot see an object, that number would exponentially rise.

VII. Conclusion

In this paper, we have presented a method of determining the 6D position of an object based on singular value decomposition and correlation of gradients in log-polar space. PCA allowed us to accurately detect an object, while using the same method with a set of HOG features allowed us to give an initial estimate of the object's orientation. Afterwards, we have taken the gradients of the grayscale images of the estimates and transformed them to log-polar space. The properties of the log-polar space, scale and rotation invariance, have allowed us to lower the number of matches (number of reference images) to compare, reducing computation time. Moreover, we have taken only part of the log-polar images, further reducing computation time. Finally, using iterations assures operation in real-time, while the results on Fig. 6 show that the algorithm will converge with accurate results.

Acknowledgment

The results presented in the paper are obtained in the scope of the research projects: “Development and Experimental Performance Verification of Mobile Dual-Arms Robot for Collaborative Work with Humans”, Science and Development Program – Joint Funding of R&D Projects of the Republic of Serbia and the People’s Republic of China, contract no. 401-00-00589/2018-09, 2018-2021 and national R&D project no. TR-35003, both supported by the Ministry of education, science and technology development of the Republic of Serbia.

References

[1] Labbé, Y., Carpentier, J., Aubry, M., & Sivic, J. (2020, August). “CosyPose: Consistent multi-view multi-object 6D pose estimation”. In *European Conference on Computer Vision* (pp. 574-591). Springer, Cham.

[2] Luis Malagón-Borja, Olac Fuentes, “Object detection using image reconstruction with PCA”, *Image and Vision Computing, 27 (1-2), 2-9*

[3] Huang, S. C., Huang, W. L., Lu, Y. C., Tsai, M. H., Lin, I. C., Lau, Y. C., & Liu, H. H. (2019, January). “Efficient Recognition and 6D Pose Tracking of Markerless Objects with RGB-D and Motion Sensors on Mobile Devices”. In *VISIGRAPP (1: GRAPP)* (pp. 375-382).

[4] H. S. Prasantha, H. L. Shashidhara and K. N. Balasubramanya Murthy, "Image Compression Using SVD," International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007), 2007, pp. 143-145

[5] R. Matungka, Y. F. Zheng and R. L. Ewing, "2D invariant object recognition using Log-Polar transform," 2008 7th World Congress on Intelligent Control and Automation, 2008, pp. 223-228