Using OpenCV for real-time image recognition through augmented reality devices

Gabdullina Dinara
Kazan Federal University, Kazan, Russian Federation, 420008, Rep. Tatarstan, Kazan, St. Kremlin, 35, Russia

Zykov Evgeniy
Kazan Federal University, Kazan, Russian Federation, 420008, Rep. Tatarstan, Kazan, St. Kremlin, 35, Russia

Kugurakova Vlada
Kazan Federal University, Kazan, Russian Federation, 420008, Rep. Tatarstan, Kazan, St. Kremlin, 35, Russia
E-mail: dinaragabdullina086@gmail.com, Evgeniy.Zykov@kpfu.ru, vlada.kugurakova@gmail.com

Abstract:

This article describes the peculiarities of writing libraries and the rapid development of augmented reality applications on the Unity platform and the OpenCV open-source library. The application of OpenCV functions to work with augmented reality objects is discussed in detail. On their basis, algorithms and methods were developed and an open library was created that implements the stable recognition of objects in various conditions by means of binding to the marker and their visualization on the augmented reality devices. On the basis of the developed library, a complete application was created to illustrate the possibility of applying virtual reality technologies for optical control of the assembly of radio electronic boards and for personnel training.

Keywords: Augmented reality, Pattern recognition, Computer vision, Optical control, Industrial automation.

1. Introduction

Augmented reality is a good tool for implementation in various industries, but the prevalence of the use of augmented reality applications in industry is still low. The one article says [1], that the reasons for the non-acceptance of new technologies are described: risk factors, which include time risks, financial and security risks, and the factor of psychological acceptance of the technology.

The list of successful examples of the use of augmented reality in industrial processes could go on and on. But so far, every augmented reality application is a "one-off product", so there is a need to streamline the development of augmented reality applications, easily using different glasses and relying on proven approaches to sustainably recognize dynamically changing 3D objects in different lighting conditions, expanding the range of applications in industrial processes – such as conveyor assembly.

However, in order to solve all these problems a considerable amount of work needs to be done, so it was decided to start by narrowing down the scope of the tasks to be solved, with possible scaling up by adding more functions and algorithms.

So, the task was to create an open library of sustainable object recognition in different light conditions, visualizing them and other related information on augmented reality devices based on freely distributed software.

2. SDK analysis for AR

ARKit [2] is an SDK for creating augmented reality applications for the IOS operating system. Available features include environmental depth estimation using the Depth API and LiDAR camera, capturing human

motion with the camera and identifying human bones, and creating topological space maps.

ARCore [2] allows the development of Android and iOS applications. The depth of the environment can be measured with an RGB camera by creating a depth map. There is a light estimation function, marker recognition.

The Vuforia augmented reality SDK [2], developed by Qualcomm, provides capabilities for creating Android and iOS applications. With Vuforia, you can recognize objects from a 3D model, scan objects in space, and recognize groups.

Wikitude [2] is another SDK, object recognition from 3D models, image recognition and tracking, point-to-point mapping using location, video overlay, smartglass integration, integration with external plugins.

3. Problem statement. Description of the basic functionality for an augmented reality library on the programmer and user side

Let's describe the basic functionality with an example: you need to recognize an object in space. It was started by defining the user's actions:

1. The user opens the application.
2. The user points the camera of the glasses at the workspace.
3. A frame appears around the object of interest, indicating that recognition is correct. You will be prompted for further action.
4. Repeat steps 2-3 until the work is completed. Close the application when the work is completed.

Let's describe the basic functionality on the developer side:

1. Login to the app and initialize the augmented reality device.
2. The algorithm recognizes key points in the image and in the frame.
3. Finding correspondences between the key points.
4. Drawing a frame around the object.
5. Exit the app.

3.1 Image recognition with OpenCV

OpenCV is an Open-Source Computer Vision Library. The library is quite popular, currently having over 5 million downloads, and is the baseline for the Khronos computer vision standard. The library is adapted to different platforms. Independence from closed code of various commercial frameworks and possibility to implement new algorithmic approaches are the main advantages of using OpenCV.

The OpenCVSharp plus Unity resource kit [3] and Unity 2021.2.10f1 development environment [4] were used to implement this functionality.

ORB is an algorithm for identifying key points in an image. According to [5], this algorithm is the best algorithm for low performance devices. FAST algorithm [6] is used to select the key points: a pixel with 16 pixels is compared, if there is more than half of the pixels darker or brighter than this one, it is selected as the key point. A binary feature vector is used to describe the key points using the BRIEF algorithm [7]. To begin with, smoothing is performed, then a pair of pixels is selected around the key point and the intensities are compared. Based on the result, a value of 0 or 1 is selected.

The Flann Matcher algorithm [8] was used to match the points from the image and the camera frame. Unlike the Brute-Force Matcher algorithm [9], this algorithm is faster on large data sets because it is an implementation of the k-d-tree algorithm [10]. The complexity of the algorithm is at best $O(h)$, where h is the height of the tree.

The Lowe's Ratio Test algorithm [11] was used to remove key points that are noise. In the next step, a frame was mapped around the recognized image. In order to map the corresponding points from the image to the frame, a homography, which is a 3×3 matrix, was used.

The algorithm pseudocode shown in Fig. 1 and Fig. 2, shows the steps described earlier.

```
FUNCTION object recognition in an image
    INPUT:image key points, image descriptors, frame;
    OUTPUT:frame;


Searching for frame key points and descriptors
Comparison of picture and frame descriptors
Ratio Test
Searching for homography
IF homography exists AND matched descriptors > constant
    Drawing a frame around the object on the frame
ENDIF
ENDFUNCTION
```

Fig. 1. Pseudocode to recognize key points on a frame and display the frame

```
FUNCTION Ratio Test
    INPUT:matched descriptors;
    OUTPUT:array of matches;


FOR every match
    IF distance to first best point > distance to the
    second best point multiplied by a constant THEN
        Adding the first best point to the array of
        matches
    ENDIF
ENDFOR
ENDFUNCTION
```

Fig. 2. Pseudocode for deleting keypoints

Recognition of key points in the image is performed only once, at the beginning of work, and recognition of

key points in the frame is performed on each frame from the camera. It should be said that the correctness of the frame display depends on environmental factors, so the frame is drawn only if more than 20 points are found when Lowe's Ratio Test is applied.

The app was run on the Vuzix M400 Starter Kit augmented reality glasses.

3.2 Improving the algorithm

After running several tests, the weaknesses of the developed algorithm were revealed. Firstly, the frame was not always displayed correctly on the scene. Secondly, it required a certain amount of time and good lighting to recognize a sufficient number of key points. In addition, jitter, or camera shake led to the loss of an object followed by the initiation of a new search, resulting in a chaotic flicker on the screen.

Thus, the following code improvements have been suggested:

1. Stabilization of the frame display;
2. Image pre-processing:
   2.1. bringing it into greyscale format,
   2.2. overlay filters to get rid of noise and sharpen the image;
3. Experiment with different levels of light in the workplace and choose the most appropriate filters.

In order to stabilize the frame display, it was decided to draw a single object, Cv2.Rectangle [12].

A conditional operator has also been added: if after Lowe's ratio test the matched points are less than the set threshold value, in this case 30, and at the same time greater than 20, the frame coordinates are not recalculated, but the previous ones are taken. In the other case, it is updated according to the code. If the homography is not counted, then the average value of the last 10 frames is searched using AccumulateWeighted [13].

3.3 Implementing the functionality in the application

Following this research, the functionality has been implemented in an application to assist the technician in the lead assembly of radio components on a printed circuit board (Fig. 3) [14].

The advantages of manual assembly are:

- High flexibility when changing production sites.
- The possibility of a permanent visual check, allowing defects in boards or components to be detected in good time.

The disadvantages are: low productivity, high labour intensity of the technological process, use of sufficiently qualified operating personnel (3rd-4th categories) with the necessary practical and theoretical knowledge [15].

The use of automated optical inspection systems allows higher inspection confidence and minimizes defects at high assembly speeds.

There are few companies in the Russian electronics market that produce products in large batches. This makes the development of low-cost optical inspection systems an urgent task.

The basic requirements for budget optical inspection systems are [16]:

- identifying the main types of defects – missing, misaligned, misaligned components, jumpering and lack of contact in solder joints,
- a visual presentation of information on the locations of suspected defects,
- low cost.

Based on these requirements, it is possible to formulate a number of tasks that need to be solved when creating budget optical inspection systems [12]:

- obtain an image of the reference object and obtain an image of the object under test,
- perform an initial processing of the acquired images,
- identify the presence of defects on the object under test and display a message on the type and location of the suspected defect.

On the basis of the above, the application contains the following functionality: the recognition of the board and the radio components, the display of the name, the quantity, the position on the board and the polarity. Also, if you look at the example instructions [17], the resistance or capacitance of the radio component is also indicated. This is the basic information required for a correct assembly.

Other article [18] says that the assembly procedure is covered. The main points that can be highlighted are checking the polarity of the part, fitting the part to the board, trimming the pins back and starting soldering, once all the parts have been fitted.

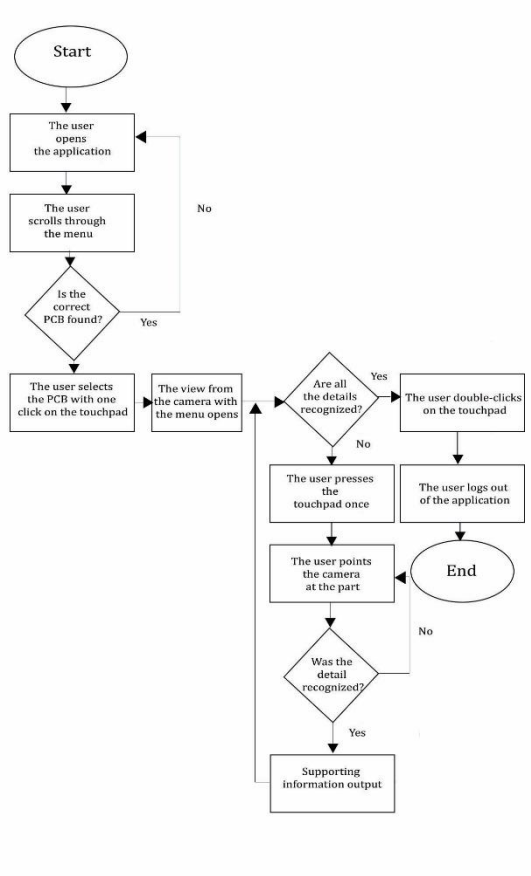

Fig. 3. Application flowchart

A mask in computer vision is used to highlight pixels with specific values in an image [19]. The use of a mask for colour recognition of radio components is a solution to the problem as these radio components do not have a lot of inscriptions or drawings on the surface. Let's take the red diode as an example.

The HSV colour extremes were first set to (155, 25, 111) and (180, 255, 255). Each frame was then converted to HSV format. To filter specific colours, the InRange function [20] was used to feed the boundary colour values to form a mask. Function Dilate [21] was used to expand mask – increase white area of the mask and get rid of possible noise. The mask was overlaid on the frame using BitwiseAnd [22].

The Cv2.FindContours method [23] was used in OpenCV to select object contours on the frame. The method takes an image on which you want to find contours. To save memory, we used the method CHAIN_APPROX_SIMPLE, which returns only the outermost points at the outline. These reference points were used to draw a rectangle in the outline area.

Different radio components have their own polarity characteristics. If the polarity is not correct, the radio element may fail. To avoid this, the user is prompted in the top panel to check the polarity [24]. There can be several identical radio components, so the number is also displayed.

As all code was implemented in C# in Unity, the interface was also created in the same environment. Due to the fact that the screen size for a single eye and its resolution is quite small, the interface should be as simple as possible, with simplified UI elements and no complex hierarchy. At the same time, only the information that the user needs at a given step should be displayed [25]. The interface is controlled by the touchpad using the Key Codes function [26].

4. Image pre-processing

The introduction of filters and lighting experiments was decided to be carried out in the last phase of the implementation, in order to obtain the most representative experimental results.

Using different filters for image processing allows you to extract additional information needed to correctly identify key points. Consider the following image processing methods: grayscale conversion, Gaussian blur, and histogram alignment.

The image in OpenCV is converted to greyscale format [27] using a formula:

$$RGB\ to\ Gray: \qquad (1)$$
$$Y \leftarrow 0{,}299 * R + 0{,}587 * G + 0{,}114 * B,$$

where Y – is the result of the conversion, *R is* the red channel, *G* – green channel, *B* – blue channel.

A conversion is made from a three-channel image to a single-channel image with values in the range 0 to 255.

Smoothing by mean value is one of the simplest smoothing methods. This method takes an area of pixels of a certain size and finds the average value for the pixel value you are looking for [21].

To determine what size the counting area of the mean will be, a kernel is used, which is a matrix with dimensions $M \times N$, where M and N - are odd numbers [28]. Also, the larger the dimensionality of the kernel, the stronger the smoothing. Example of kernel (Eq. 2):

$$K = \frac{1}{K_{height} * K_{width}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \qquad (2)$$

where K – 3×3 core.

Gaussian antialiasing is used to smooth out sharp corners and get rid of noise. This smoothing is based on

the Gaussian function formula in two-dimensional space. It is applied to each pixel, after which a convolution matrix is formed. The pixel value is updated using a weighted average formula based on the kernel dimension.

Impulse noise, or randomly occurring black and white pixels, is a type of noise in images. For this case, a median filter is used and a kernel is also used, but the median in a given vicinity rather than the mean value is considered [21].

Histogram alignment is a method of increasing the intensity range of an image. This operation is performed by overriding the pixel value using a cumulative distribution function [29]. If, however, there are heavily overexposed parts of the image, conventional histogram equalization will lose information in these areas due to the relatively large range of variation. Adaptive histogram equalization helps to solve this problem by applying the histogram equalization operation to parts of the image separately [30].

5. Introducing image preprocessing

The recognized image and the camera image were processed in the following order: conversion to grayscale, Gaussian smoothing and adaptive histogram equalization. There are corresponding methods in OpenCV: Cv2.CvtColor(), Cv2.GaussianBlur(), Cv2.Blur(), Cv2.MedianBlur, CLAHE.Apply().

A comparison was made between the recognition times before and after filter processing. This experiment was due to the fact that the developed application runs in real time. Experiments were conducted with different degrees of light, the time of correct recognition was considered the moment when the dialog prompt appeared in the upper panel. The time was measured using the StopWatch class functions: Start() and Stop.

The degree of illumination was measured using a Testo 540 luxmeter. To determine the mean, medians were chosen as the most resistant to experimental emissions values [31].

The graph (Fig. 4) shows that the recognition is much faster when image preprocessing is used, and the lower the illumination, the higher the effect of preprocessing on the object recognition time.

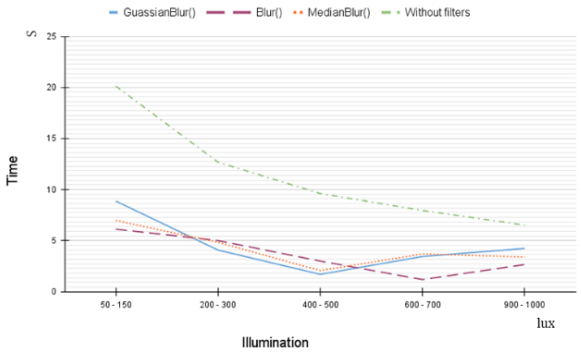

Fig. 4: Graphs without processing and processing by low-pass filters with a kernel

6. Conclusion

The algorithms of the OpenCV library allow the creation of basic functionality for augmented reality applications. Various combinations of image preprocessing, keypoint recognition and matching methods produce results that can be used for a variety of purposes. The algorithms considered combine to provide image recognition functionality in space.

The application also revealed weaknesses such as the dependence on lighting quality and on the speed of the software to display the frame on each frame. Applying pre-filtering and remembering previous values has greatly improved the display. Thus, proper use of OpenCV provides the opportunity to design a variety of functionality.

Acknowledgements

The work was carried out at the expense of the Strategic Academic Leadership Program of Kazan (Volga Region) Federal University ("PRIORITY 2030").

References

1. K.E. Schein, P.A. Rauschnabel, “Augmented Reality in Manufacturing: Exploring Workers' Perceptions of Barriers”, IEEE Transactions on Engineering Management, pp. 1-14, 2021.
2. Y. Chen, Q. Wang, “An H.C. Overview of augmented reality technology”, Journal of Physics: Conference Series, 2019.
3. OpenCV plus Unity (2019). (Accessed: 2 February 2022).
4. Unity 2021.2.10 (2021). (Accessed: 5 February 2022).

5. A. Muryan, Y.Verdi, “Application of oriented fast and rotated brief (orb) and bruteforce hamming in library opencv for classification of plants”, JISAMAR (Journal of Information System, Applied, Management, Accounting and Research), pp. 51-59, 2020.
6. P. Chhabra, N.K. Garg, M. Kumar, “Content-based image retrieval system using ORB and SIFT features”, Neural Computing and Applications, pp. 2725-2733, 2018.
7. F. Siddiqui, S. Zafar, S. Khan, N. Iftekhar, “Computer Vision Analysis of BRIEF and ORB Feature Detection Algorithms”, Applied Computational Technologies, pp. 425-433, 2022.
8. V. Vijayan, P. Kp, “FLANN Based Matching with SIFT Descriptors for Drowsy Features Extraction”, Fifth International Conference on Image Information Processing (ICIIP), pp. 600-605, 2019.
9. N. Antony, B.R. Devassy, “Implementation of Image/Video Copy- Move Forgery Detection Using Brute-Force Matching”, International Conference on Trends in Electronics and Informatics (ICOEI), pp. 1085-1090, 2018.
10. Y. Chen, L. Zhou, Y. Tang, “Fast neighbor search by using revised k-d tree”, Informatics and Computer Science Intelligent Systems Applications, pp. 142-162, 2019.
11. D.G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints”, International journal of computer vision, Vol. 60. pp. 91-110, 2004.
12. Drawing functions in OpenCV (2022). (Accessed: 20 March 2022).
13. E. Niloy, J. Meghna, M. Shahriar “Hand Gesture-Based Character Recognition Using OpenCV and Deep Learning”, International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI), pp. 1-5, 2021.
14. PCB Mounting Output Process (2022). Accessed: 25 March 2022).
15. V.L. Lanin, “Assembly technology, installation and control in the production of electronic equipment”, Bel. State University of Informatics and Radioelectronics. – Minsk: Inpredo. p. 64, 1997. (In Russian).
16. A. Ovchinnikov, “The concept of building budget systems for optical inspection of the quality of printed circuit board assembly”, Technologies in the electronic industry, Vol. 8, No. 36, pp. 41-44, 2009. (In Russian).
17. Soldering kit (2022). (Accessed: 25 March 2022).
18. A.I. Tsukanov, O.V Kuchevasov, “Technologies for mounting and dismantling components and elements of radio-electronic and radio-television equipment: a teaching aid”, St. Petersburg State Educational Institution ", College of Electronics and Instrument Engineering" St. Petersburg, p. 105, 2017.
19. OpenCV – Invert mask (2021). (Accessed: 22nd April 2022).
20. A. Qashlim, B. Basri, H. Haeruddin, A. Ardan, “Smartphone Technology Applications for Milkfish Image Segmentation Using OpenCV Library”, International Journal of Interactive Mobile Technologies (iJIM), pp. 150-163, 2020.
21. H. Singh, “Advanced Image Processing Using OpenCV”, Practical Machine Learning and Image Processing, pp. 63-88, 2019.
22. M.A. Mir, F. Qazi, M. Naseem “Invisibility Cloak using Color Extraction and Image Segmentation with OpenCV”, Global Conference on Wireless and Optical Technologies (GCWOT), pp. 1-6, 2022.
23. V. Harini, V. Prahelika, I. Sneka, “Hand Gesture Recognition Using OpenCv and Python”, New Trends in Computational Vision and Bio-inspired Computing, pp. 1711-1719, 2018.
24. Capacitor Polarity: Understanding Polarity (2022). (Accessed: 12 March 2022).
25. UI Best Design Practices (2022). (Accessed: 23 March 2022).
26. How to map Android keycode to Unity keycode (2021). (Accessed: 23 March 2022).
27. W. Hou, D. Xia, H. Jung “Video road vehicle detection and tracking based on OpenCV”, International Conference on Information Science and Education (ICISE-IE), pp. 315-318, 2020.
28. Convolutions with OpenCV and Python (2016). (Accessed: 24 March 2022).
29. Y. Wei, G. Xu, H. Liu “OpenCV-based 3D printing physical surface defect detection”, IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)”, pp. 1924-1927, 2020.
30. F. Alkhalid, A.M. Hasan, A. Alhamady, “Improving radiographic image contrast using multi layers of histogram equalization technique”, IAES International Journal of Artificial Intelligence (IJ-AI), pp. 151-156, 2021.

31. Median in statistics (2022). (Accessed: 1 May 2022).

Authors Introduction

Ms. Gabdullina Dinara

Laboratory assistant, Research Laboratory of Augmented Reality Technologies in Industrial Processes, Kazan Federal University. Research area: augmented reality, virtual reality, computer vision

Mr. Zykov Evgeniy

Candidate of Science in Physics and Mathematics, Associate Professor, Department of Radio Astronomy, Kazan Federal University. Research area: image processing, artificial intelligence, neural networks, robotics, augmented reality, bionics.

Ms. Kugurakova Vlada

Candidate of Technical Sciences, Associate Professor of the Department of Software Engineering, Kazan Federal University. Research area: virtual reality, augmented reality, 3D models, game engines, artificial intelligence.