IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified ⁜ Impact Factor 8.066 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 10, Issue 7, July 2023

DOI: 10.17148/IARJSET.2023.107128

A New API for Computer Vision Based Augmented Reality

Ms Vinaya Hande[1], Ms Sheetal Bandekar[2]

MCA Student, Master of Computer Applications, KLS Gogte Institute of Technology, Belagavi, India[1]

Assistant Professor, Master of Computer Applications, KLS Gogte Institute of Technology, Belagavi, India[2]

Abstract: The paper describes research contribution towards Programming Augmented Reality API using open-source software tools such as Python and OpenCV. This API inputs an image scans and describes the same for unique features, match the features with user input image and then overlay the desired image over the existing image. All the work was created in Python programming language utilizing Pycharm IDE, OpenCV and Numpy Libraries.

Keywords: Computer Vision, Image Detection, Feature Matching, Augmented Reality, OpenCV, Numpy, ORB Algorithm, Homography Matrix and FLANN's Algorithm.

I. INTRODUCTION

In the early 1960s, the emergence of Augmented Reality (AR) was a direct outcome of advancements in computer technologies, including faster processors, real-time rendering methods, and the evolution of position tracking and artificial vision systems. These innovations synergized to create applications capable of overlaying real-time video captured by cameras with images, 3D models, texts, and other digital components.

Since its inception, AR has been recognized as a paradigm that seamlessly merges the physical environment with digital elements, generating a dynamic real-time composite view. This integration facilitates remarkably natural interactions due to the employment of artificial vision techniques, enabling users to easily detect and engage in simple interactions. As the pervasive presence of digital information continues to shape our daily routines, the boundaries between the tangible and virtual realms gradually blur.

AR essentially entails the integration of digital data into the real world. Leveraging AR technologies, digital information can be seamlessly incorporated into the user's immediate surroundings, eliminating the need for the user to divert explicit attention to a device screen. Unlike other interaction models, AR preserves users' connection to their real-world contexts, ensuring their focus remains on the tangible environment. This results in the creation of an augmented real world without isolating the context. Capitalizing on the user's visual and spatial capabilities, AR enriches the real world by providing supplementary information, rather than immersing the user within a computer-generated virtual realm. [1]

Paul Milgram and Fumio Kishino introduced the concept of the "virtuality continuum." This notion outlines a spectrum spanning from actual reality to computer-generated virtual reality. Situated within this continuum is a subset termed mixed reality, encompassing the entire spectrum between tangible reality and a completely virtual environment. (See Figure 1).

Fig. 1 Virtual continuum

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified ⁜ Impact Factor 8.066 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 10, Issue 7, July 2023

DOI: 10.17148/IARJSET.2023.107128

These days the computers include a lot of learning ability that has a coordinated webcam, the necessity of clear and reasonable images brought humanity to make techniques to enhance a picture, if important decrease commotion, obscure, modify the splendour, shading transformation, and so on. Our work builds up a program to identify an Image, detect it features and match the same and then augment desired expanded reality on a screen.

Here ORB (Oriented FAST and Rotated BRIEF) to detect and match features of a particular image.

II. TECHNOLOGY USED

The remaining sections of this paper are composed as follows. Section 2.1 briefs about the technologies utilized in the implementation of algorithm. Section 3 gives the steps in implementation. Finally, Section 4 gives conclusion.

A. Computer Vision

Computer Vision constitutes a multidisciplinary domain focused on enabling computers to attain a sophisticated comprehension of digital images and videos. It strives to automate functions akin to those executed by the human visual system.

This encompasses a sequence of actions encompassing the capture, manipulation, scrutiny, and interpretation of digital images, leading to the extraction of intricate data from the physical environment to generate numerical or symbolic insights. [2][5]. Key functions encompassed within the realm of Computer Vision in Python encompass:

- Recognition: Object recognition involves the task of discerning and pinpointing particular items within an image or a stream of video. This can include faces, vehicles, animals, and Text Recognition (OCR - Optical Character Recognition) includes Extracting text from images and converting it into machine-readable text.
- Analysis of Motion: Optical Flow calculates the motion of entities between successive frames in a video, while Object Tracking involves tracing the trajectory of particular objects across these frames and Gesture Recognition detects and interprets the gestures made by human hands or body parts.
- Reconstruction of Scenes: 3D Reconstruction creates a 3D model of a scene or object from multiple 2D images or video frames and Depth Estimation does inferring the depth information of a scene or objects from 2D images or a stereo camera setup.
- Restoration of Images: DE noising used for removing noise from images, which can be caused by various factors like low-light conditions or sensor limitations, DE blurring Reduces or eliminates the blur caused by camera shake or out-of-focus capture and Super-Resolution Enhances the resolution and quality of an image to obtain a higher- resolution version.

B. OpenCV Python Computer Vision

OpenCV-Python serves as a bridge between Python and the OpenCV C++ API, enabling the combined potential of both. Designed to address computer vision challenges, it provides Python bindings, utilizing NumPy to seamlessly translate array structures. This compatibility extends to integration with complementary libraries like SciPy and Matplotlib, which are reliant on NumPy.

C. Augmented Reality

Augmented reality transforms real-world settings by incorporating computer-generated techniques that continually enhance the environment. This augmentation usually involves a blend of visual, auditory, and tactile/haptic interactions to enrich the overall experience. Components of Augmented Reality are

Camera or Sensor: An AR system requires a camera or sensors to capture the real-world environment and provide input to the AR software.

Computer Vision: Utilizing computer vision algorithms, objects and characteristics within the real-world environment can be identified and monitored. These algorithms help in overlaying virtual objects accurately onto the physical world. Tracking: AR systems use tracking techniques to understand the position and orientation of the camera or the device relative to the physical world. This information is critical for placing virtual objects in the correct locations. [1] Rendering: The AR software generates virtual objects or information, which are then superimposed onto the camera feed. These objects are rendered in real-time and adjusted based on the user's perspective.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified ⁜ Impact Factor 8.066 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 10, Issue 7, July 2023

DOI: 10.17148/IARJSET.2023.107128

III. METHODOLOGY USED

This API contains Image Detection, Feature Matching and Augmented Reality with Open CV Module and Numpy Module using ORB Algorithm, Homography Matrix and FLANN's Algorithm

A. Software Components

The software components for the development of the API are:

Python: General Python programming allows for code creation that can run on popular operating systems such as Linux, Mac OSX, and Windows. In this instance, the development work was conducted within a Windows environment. OpenCV provides various functions for image processing and computer vision tasks. For example, you can perform operations like resizing, converting colour spaces, applying filters, detecting edges, and more. The combination of Python's ease of use and OpenCV's capabilities makes it a popular choice for computer vision and image processing projects.

- OpenCV: OpenCV could be a library of programming capacities mainly geared toward real time computer vision, created by Intel Russian exploration, was intended for computational effectiveness, the library can make the most of the equipment increasing speed of the basic heterogeneous process stage.OpenCV has a large and active community of developers, and it is widely used in various fields, including robotics, computer vision research, medical imaging, automotive applications, augmented reality, and more. As OpenCV supports multiple programming languages, it is highly accessible to developers with different language preferences. Python is one of the most popular languages used with OpenCV due to its simplicity and extensive support in the computer vision and data science communities. The Python binding for OpenCV provides an easy-to-use and expressive interface, making it an excellent choice for rapid prototyping and development of computer vision applications.

- Numpy: NumPy stands as the essential cornerstone for scientific computation within Python. This library furnishes a versatile array entity, alongside related constructs like masked arrays and matrices. It offers a collection of efficient functions for rapid array operations, encompassing mathematical, logical, shape manipulation, sorting, selection, input/output, discrete Fourier transforms, elementary linear algebra, fundamental statistical calculations, random simulations, and numerous other functionalities. At the heart of NumPy lies the ndarray object, which serves as the nucleus. This entity encapsulates n-dimensional arrays hosting uniform data types, with a multitude of operations executed via compiled code to optimize performance. Numpy simplifies tasks by offering a MATLAB-style syntax and serving as a highly optimized library for numerical operations. It seamlessly converts between OpenCV array structures and Numpy arrays, facilitating smooth interchangeability. Moreover, Numpy operations can be seamlessly integrated with OpenCV functionalities.

B. Algorithms

Algorithms used for the development of the API are:

- ORB algorithm: ORB is an amalgamation of the FAST keypoint detector and BRIEF descriptor, augmented with supplementary attributes to enhance its effectiveness. FAST, known as Features from Accelerated Segment Test, is employed to identify features within the given image and employs a pyramid for generating features at multiple scales. However, FAST does not calculate the orientation and descriptors for these features, which is where BRIEF steps in to fulfill this role**.**

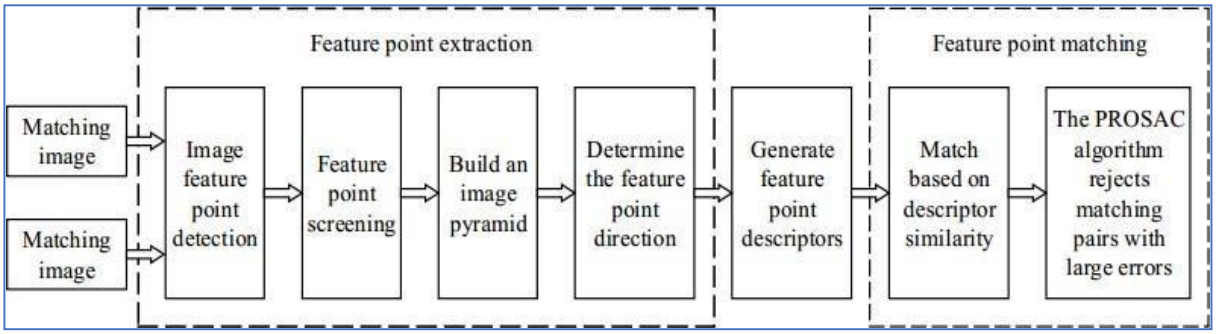

Fig. 2 *Image* matching flow chart based on ORB algorithm

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified ⁜ Impact Factor 8.066 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 10, Issue 7, July 2023

DOI: 10.17148/IARJSET.2023.107128

ORB integrates BRIEF descriptors; however, due to BRIEF's susceptibility to perform inadequately under rotations, ORB takes a strategic approach. It aligns the BRIEF descriptor with the key point's orientation by deriving a rotation matrix from the patch's orientation. This rotational adjustment ensures a well-oriented BRIEF representation. As a whole, ORB emerges as a proficient substitute for feature extraction, offering advantages over SIFT or SURF algorithms in terms of computational efficiency, matching accuracy, and particularly patent-related considerations.
Below is a code which shows the use of ORB

```
          import numpy as np
import cv2from matplotb import pyplot
               as plt

  img = cv2.imread('simple.jpg',0)

      # Initiate STAR detector
          orb = cv2.ORB()

    # find the keypoints with ORB
     kp = orb. Detect (img, None)

 # compute the descriptors with ORB
   kp, des = orb.compute(img, kp)

 # draw only keypoints location,not
        size and orientation
                img2 =
cv2.drawKeypoints(img,kp,color=(0,255
           ,0), flags=0)
     plt.imshow(img2),plt.show()
```

Fig. 3 ORB Algorithm Code

Following is the output of above code (Fig. 3)

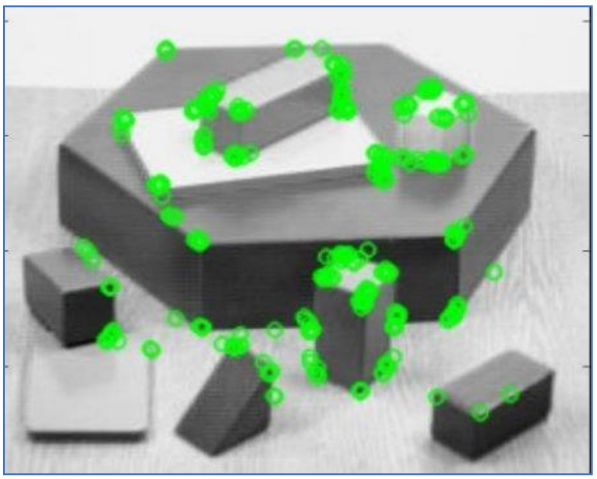

Fig. 4 Output of ORB algorithm

- FLANN's Algorithm: FLANN, an acronym for Fast Library for Approximate Nearest Neighbours, encompasses an array of algorithms meticulously designed to expedite the quest for the nearest neighbours within extensive datasets and in scenarios involving high-dimensional attributes. Particularly adept at handling substantial data, FLANN outpaces BFMatcher for large datasets. To illustrate with a case involving the FLANN-based matcher, the procedure necessitates the utilization of two dictionaries. The initial dictionary, IndexParams, defines the chosen algorithm along with relevant parameters. The subsequent dictionary, SearchParams, outlines the extent to which the index trees should be traversed recursively. Opting for higher values amplifies precision but concurrently extends processing time. [5][2]

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified ⁜ Impact Factor 8.066 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 10, Issue 7, July 2023

DOI: 10.17148/IARJSET.2023.107128

Below is output generated by FLANN's algorithm (Figure 4)

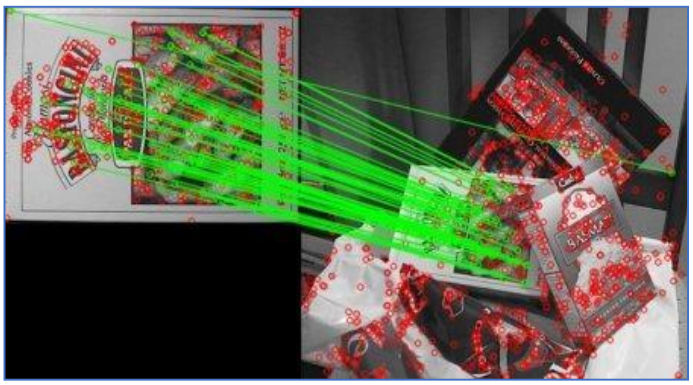

Fig. 5 Output generated by FLANN's algorithm

- Homography Matrix: In the realm of computer vision, a fundamental principle dictates that any pair of images capturing the same flat surface within three-dimensional space exhibit a homographic relationship, assuming the application of a pinhole camera model. This principle yields numerous practical utilities, including image rectification, image alignment, and the determination of camera movement—comprising both rotation and translation—between these images. Extracting camera rotation and translation information from an inferred homography matrix serves as a valuable asset, finding application in navigation or the seamless integration of 3D object models into images or videos. This integration ensures correct perspective rendering, seamlessly blending the objects with the original scene. Once the homography matrix is computed, it can be used to transform points or images from one coordinate system to another, effectively aligning the two images. This alteration constitutes a projective transformation, enabling accommodation of perspective alterations, rotations, translations, and scaling. Estimating the homography can be achieved through methods like the Direct Linear Transform (DLT) algorithm (refer to source 1 for further details). Considering the object's planar nature, the conversion between object frame points and projected image plane points within the normalized camera frame is accurately captured through a homographic relationship.
-

Briefly, the planar homography relates the transformation between two planes (Figure. 5)

$$s\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

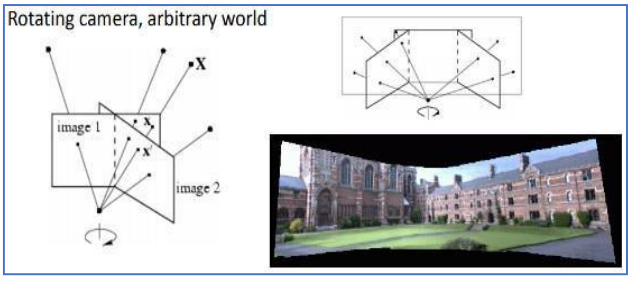

Fig. 6 A camera rotating around its projection axis can be likened to treating the points as if they lie on an infinitely distant plane. [3][4]

C. New API

This API uses completely open source python libraries to build computer vision project with inputting an image, detecting and matching the features of the given image and project another image on the same. This API gives complete control of Object Detecting, Object Matching and Augmented Reality. One can define how many features to be detected and matched and can easily be incorporated in other projects

- Feature comparison of new API with existing

Although the realm of instant Augmented Reality (AR) applications can be explored using pre-built libraries, the

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified ⁜ Impact Factor 8.066 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 10, Issue 7, July 2023

DOI: 10.17148/IARJSET.2023.107128

incorporation of Python for AR, though relatively emerging, opens up a realm of vast possibilities. Essentially, AR can be perceived as a variation of the Computer Vision challenge, a domain Python has effectively addressed through diverse libraries like OpenCV, rendering it an apparent choice for AR app development. Admittedly, existing libraries (such as ARCore, primarily coded in languages like C++ or C#) offer comprehensive solutions, enabling developers to swiftly introduce virtual elements into reality with minimal coding effort. Nevertheless, harnessing Python's state-of-the-art capabilities empowers developers with complete autonomy over the process of projecting virtual objects into the real world. This includes pivotal tasks like feature extraction, homography, projection, and other crucial aspects, facilitating independent control over every facet of the AR creation process.

IV. RESULTS AND OBSERVATION

First let us provide an image to detect and describe features using ORB algorithm. The image is converted to grey scale to detect maximum features. Here's the output of Feature detection, matching and Augmented Reality with ORB algorithm, Flann's Algorithm and Homography Matrix:

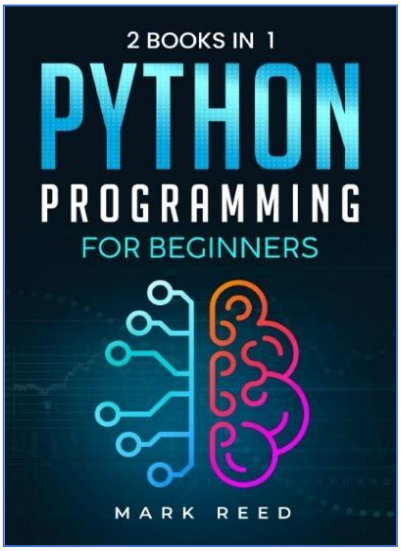

Fig. 7 Input image

Here's the output of Feature detection with ORB algorithm:

Fig. 8 ORB Feature Detection

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified ⁜ Impact Factor 8.066 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 10, Issue 7, July 2023

DOI: 10.17148/IARJSET.2023.107128

Here's the output of Feature detection and matching with Flann's algorithm:

Fig. 9 Flann's algorithm feature matching

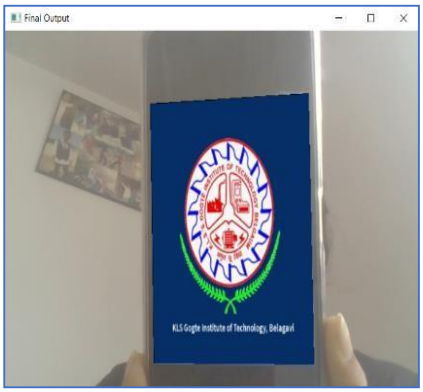

Fig. 10 Output using new API

V. CONCLUSION

This API can be incorporated as well as customized according to user in other projects. OpenCV is the best free device for character acknowledgment yet it is exceptionally delicate to the commotion. An image should have prominent features. One can make their own AR API by using Open Source libraries such as ORB, Numpy. Even if the object is tilted the augmented image is perfectly projected with help of homography matrix.

REFERENCES

[1]. Wei, H.S., Huang, W.J., Dong, Q., Liu, Y.L. (2019) Shadow detection of outdoor video from the mobile perspective of augmented reality. J. Journal of computer aided design and graphics,31(06):997-1006.

[2]. Dahliyusmanto, M. Nizam, D.W. Anggara, E. Hamdani, B. Anto, Z. Laili, "Marker Quality Improvement in Augmented Reality Using Grey- Scale Method", International Journal of Electrical, Energy and Power System Engineering, 2019, Vol.2, pp. 17–23.

[3]. Onyedinma E., I. Onyenwe, H. Inyiama, "Performance Evaluation of Histogram Equalization and Fuzzy Image Enhancement Techniques on Low Contrast Images", International Journal of Computer Science and Software Engineering (IJCSSE), 2019, Vol.8, pp. 144–150.

[4]. Yang, J.; Qin, D.; Tang, H.; Bie, H.; Zhang, G.; Ma, L. Indoor Positioning on Smartphones Using Built-In Sensors and Visual Images. Micromachines 2023,14, 242.

[5]. S. Line, "Feature detection and matching," [Online]. Available: https://https://medium.com/data-breach/introduction-to-feature-detection-and-matching-65e27179885d. [Accessed: July 17, 2023].