1 Introduction

In this project, three different methods for implementing augmented reality (AR) image projection were explored using OpenCV: **ArUco marker-based tracking** and **feature-based tracking using SIFT and ORB**. Each method has unique pro's and cons, depending on application requirements like precision, speed, and environmental conditions.

2 ArUco Marker-Based Tracking

Overview: ArUco markers are predefined 2D square patterns that can be easily detected and uniquely identified in an image.

1. ArUco markers are placed on the real-world surface where the overlay will be projected.
2. Using OpenCV's `cv2.aruco.detectMarkers()` function, the application detects the markers' corners in the video frame.
3. Once the markers are detected, a homography matrix is computed using the marker corners to warp the source image onto the specified region in the frame.
4. The warped image is then blended with the frame to create an augmented reality overlay.

Pros:

- **Reliability**: ArUco markers are robust and designed for quick recognition, even in low-light or cluttered backgrounds.
- **Simplicity**: Easy to implement due to the unique patterns that simplify marker detection and corner extraction.
- **Consistency**: Accurate overlay positioning as long as markers are fully visible in the frame.

Cons:

- **Dependency on Markers**: Requires visible ArUco markers, which means the environment must be prepped with these markers.
- **Limited Flexibility**: Marker placement can be restrictive, especially in dynamic or uncontrolled environments.
- **Occlusion Sensitivity**: If markers are obscured, detection fails, interrupting AR projection.

3 Feature-Based Tracking Using SIFT (Scale-Invariant Feature Transform)

Overview: SIFT is a robust feature detection method that extracts distinct keypoints and descriptors from an image, making it useful for finding correspondences between images under various transformations.

1. Keypoints and descriptors are detected in both the source image and each video frame using SIFT.
2. A brute-force matcher (`cv2.BFMatcher`) is used to find matching descriptors between the source and frame images.
3. A homography matrix is computed from matched points using RANSAC to filter outliers.
4. The source image is warped and blended onto the frame according to the homography matrix.

Pros:

- **Robustness to Transformations**: SIFT can handle scale, rotation, and minor perspective changes, making it resilient under different viewing angles and distances.
- **No Marker Requirement**: This method can be used in markerless environments, allowing for more flexible applications.
- **Accurate Matching**: SIFT is particularly effective for complex scenes due to its highly distinctive keypoints.

Cons:

- **Computationally Intensive**: SIFT is relatively slow and computationally expensive, which can cause frame rate drops in real-time applications, especially on less powerful hardware.
- **Susceptibility to Illumination Changes**: While robust to transformations, SIFT can be impacted by significant changes in lighting or shadows.
- **Requires High-Quality Features**: For reliable performance, the source image needs distinct features. Bland or repetitive surfaces reduce the matching quality.

4 Feature-Based Tracking Using ORB (Oriented FAST and Rotated BRIEF)

Overview: ORB is a fast alternative to SIFT, offering good performance for real-time applications by using binary descriptors.

1. ORB detects keypoints and computes binary descriptors in both the source image and the video frame.
2. A brute-force matcher is used to match keypoints.
3. A homography matrix is calculated from matched points, and RANSAC is applied to remove outliers.
4. The source image is warped based on the homography matrix and blended into the frame.

Pros:

- **Fast and Efficient**: ORB is optimized for speed, making it well-suited for real-time applications where computational resources are limited.

- **Markerless Tracking**: Like SIFT, ORB works in environments without predefined markers.
- **Good for Real-Time Applications**: ORB provides an efficient balance between speed and accuracy, particularly useful for applications requiring a high frame rate.

Cons:

- **Lower Feature Match Quality**: ORB may produce less reliable feature matching compared to SIFT, especially in scenes with high complexity or texture.
- **Less Robust to Scale Variations**: While ORB is rotation invariant, it is less scale invariant, making it sensitive to changes in distance.
- **Limited Performance in Low-Feature Scenes**: Similar to SIFT, ORB's performance is reduced on scenes that lack distinct visual features.

5 Applications

5.1 Projection mapping with camera projector system

In an application where a projection is projected on a surface there are many advantages in using ArUco markers over feature detection and matching. Projection mapping requires a high degree of accuracy and consistency in tracking objects. Features rely on the environment not changing which is not the case when projecting an image. The projection obstructs the environment which makes the features detected before a projection undetectable. ArUco makers are also obstructed by the projection but because of their easily recognizable structure but the detection will be less reliable.

5.2 Self-Driving Cars

Autonomous driving uses feature detection and matching. Using ArUco markers for self-driving cars would require markers needing to be installed and maintained across all roadways. This approach is not scalable and limits adaptability since it relies on reference points. Additionally, it lacks flexibility because it does not allow the vehicle to learn from or respond to sudden environmental changes.

Feature matching can detect and recognize elements like road signs, lane markings, and buildings, which are all commonly present in the environment. This method is highly scalable and cost-effective, as it doesn't require ArUco markers to be spread and maintained. It also enables continuous learning and adaptation, helping autonomous vehicles respond dynamically to new or changing environments.

6 Conclusion

- **Aruco-Based Method**: Ideal for **fixed or controlled environments** where setup with markers is feasible and occlusion can be minimized. Suitable for applications where reliability is more critical than adaptability or flexibility.
- **SIFT-Based Method**: Best suited for **highly variable, complex scenes** where transformations like scale and rotation are common, and markerless operation is essential. However, it may not be ideal for real-time applications without high processing power due to its computational intensity.
- **ORB-Based Method**: Well-suited for **real-time applications** in markerless environments where computational resources are limited, making it ideal for mobile or embedded systems. However, it may struggle with scenes requiring high accuracy or complex transformations.

Each method can be beneficial depending on the specific requirements, making it essential to choose based on the environment, hardware constraints, and accuracy needs.