5 Point Algorithm

1. 순차적으로 입력되는 영상으로부터 ArUco marker detection을 수행하여, corners 점들과 ids들을 계산한다.
2. 첫번째 영상이 입력으로 들어가면, 동일한 과정을 수행하여 corners점들과 ids들을 계산한다.
3. 두번째 영상이 입력으로 들어가기 전, 이전에 계산한 corner점들과 ids를 저장한다.

```
prev_corners = corners
prev_ids = ids
```

4. 두번째 영상이 들어오면 marker detection을 수행하고 corner점들과 ids들을 계산한다.
5. 이전 영상에서 탐지된 Aruco Marker가 있고, 현재 영상에서도 탐지된 Marker가 있다면, 동일 ID에 대한 코너점들을 저장한다

```
if ids is not None:
  if prev_ids is not None:
    #현재 영상으로부터의 corner들과 ids들 초기화
    updated_corners = []
    updated_ids = []
    #이전 영상으로부터의 corner들과 ids들 초기화
    prev_updated_corners = []
    prev_updated_ids=[]

    for present_id in ids:
        #두 이미지에서 동일 ID를 가진 Aruco marker를 가질 경우, 그 때의 마커들의 정보를 저장
        try:
            #첫번째 코너(좌측 위)의 정보만을 저장할 경우 : [0][0]
            #코너 전체를 저장할 경우 : [0]
            index = np.where(prev_ids == present_id)[0][0]
            updated_corners.append(corners[ids.tolist().index(present_id)][0][0])
            updated_ids.append(present_id)
            prev_updated_corners.append(prev_corners[index][0][0])
            prev_updated_ids.append(prev_ids[index])
        except IndexError:
            pass
```

6. 탐지된 두 영상으로부터 5point Algorithm을 수행한다

```
updated_corners_np = np.array(updated_corners)
prev_corners_np = np.array(prev_updated_corners)
updated_corners_flat = updated_corners_np.reshape(-1, 1, 2)
prev_corners_flat = prev_corners_np.reshape(-1, 1, 2)

E, _ = cv2.findEssentialMat(prev_corners_flat, updated_corners_flat, K,
                            method=cv2.RANSAC, prob=0.999, threshold=1.0)

_, R, t, _ = cv2.recoverPose(E, prev_corners_flat, updated_corners_flat, K, distanceThresh=99999)
```

7. 위 코드에서 계산한 R,t는 각각 두 영상으로부터 계산한 회전행렬과 이동행렬이다. Visualization은 다음과 같이 알 수 있다.

```
print("Rotation Matrix:")
print(R)
print("Translation Vector:")
print(t.T)
```

8. 전체 영상으로 3번~7번 과정을 반복한다.