Homography Detection Tool – PoC

This Proof of Concept (PoC) demonstrates a Homography Detection Tool using OpenCV. Homography is used in computer vision to find a perspective transformation between two planes. It's widely applied in panorama stitching, augmented reality, and object recognition.

Python Code for Homography Detection:

```
import cv2
import numpy as np

def detect_homography(img1_path, img2_path):
    img1 = cv2.imread(img1_path, cv2.IMREAD_GRAYSCALE)
    img2 = cv2.imread(img2_path, cv2.IMREAD_GRAYSCALE)

    orb = cv2.ORB_create()
    kp1, des1 = orb.detectAndCompute(img1, None)
    kp2, des2 = orb.detectAndCompute(img2, None)

    bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
    matches = bf.match(des1, des2)
    matches = sorted(matches, key=lambda x: x.distance)

    src_pts = np.float32([kp1[m.queryIdx].pt for m in matches[:10]]).reshape(-1, 1, 2)
    dst_pts = np.float32([kp2[m.trainIdx].pt for m in matches[:10]]).reshape(-1, 1, 2)

    M, mask = cv2.findHomography(src_pts, dst_pts, cv2.RANSAC, 5.0)
    result = cv2.warpPerspective(img1, M, (img2.shape[1], img2.shape[0]))
    cv2.imshow("Homography Result", result)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

Steps Performed:

Read two images to compare.Use ORB to detect keypoints and compute descriptors.Match features using BFMatcher.Estimate the homography matrix with RANSAC.Apply the transformation and show the aligned result.