Poročilo - Vaja 2

Na spodnjih slikah lahko vidimo prikaz delovanja generatorja slik. Implementirane so metode za osnovne like ter homografske transformacije. Dodane so tudi funkcije za filtriranje odsekanih točk in barv ki so preblizu za dobro ločljivost likov od ozadja.

```python
def appy_homography_to_image(img,points):
    height, width = img.shape[:2]

    hbuffer = int(height/4)
    wbuffer = int(width/4)

    p1 = [random.randint(0, hbuffer), random.randint(0, wbuffer)]
    p2 = [random.randint(0, hbuffer), random.randint(width-wbuffer, width)]
    p3 = [random.randint(height-hbuffer, height), random.randint(width-wbuffer, width)]
    p4 = [random.randint(height-hbuffer, height), random.randint(0, wbuffer)]

    plist = [p1,p2,p3,p4]

    for i in range(random.randint(0,3)):
        tmp = plist.pop(0)
        plist.append(tmp)

    homography_matrix = cv2.getPerspectiveTransform(np.float32(plist),np.float32([[0,0],[width,0],[width,height],[0,height]]))

    img = cv2.warpPerspective(img, homography_matrix, (width,height))

    #apply homography to points
    i = 0
    l = len(points)
    while i < l:
        p = np.array([points[i][0],points[i][1],1])
        p = np.matmul(homography_matrix,p)
        p = np.array([p[0]/p[2],p[1]/p[2]])
        #check if point is NOT inside image
        if p[0] < 0 or p[0] > width or p[1] < 0 or p[1] > height:
            np.delete(points,i)
            l -= 1
        else:
            i += 1

    return img, points
```