1. Get blue point on image.

Define target coordinate and use pixel to get blue point

```
target_coord = {
    'left_hat': ((180, 52), (400, 320)),
    'right_hat': ((963, 6), (1206, 290)),
    'left_body': ((160, 350), (507, 600)),
    'right_body': ((925, 320), (1280, 590)),
    'left_foot': ((232, 600), (500, 850)),
    'right_foot': ((961, 590), (1280, 875))
}

left_pts, right_pts = [], []
for _, image in enumerate(img):
    left_pt, right_pt = fun.find_blue(image, 720, target_coord, 25)#get blue point on image
    point_lft.append(left_pt)
    point_rgt.append(right_pt)
```

2. Find corresponding point up, vp by epipolar line.

After calculate epipolar line from each uv point, select the point is colest to the line, it's the up, vp point.

```
for left_pt in point_lft[i]:
    left_pt = np.array([left_pt[0], left_pt[1], 1])
    epi=fun.epi_polar_line(F,left_pt)#calculate epipolar line
    min_dist = 150
    u, v = left_pt[0], left_pt[1]
    up=None
    vp=None
    for right_pt in point_rgt[i]:
        right_pt = np.array([right_pt[0]-720, right_pt[1], 1])
        dist = abs(np.dot(right_pt, epi))
        if dist < min_dist:
            min_dist = dist
            up=right_pt[0]
            vp=right_pt[1]
    if up == None or vp == None :#if can't find correct corresponding point, then skip
        continue
```

3. Calculate 3D point by direct triangle

After get the corresponding point, calculate the 3D point by direct triangle method, and give the RGB value from first image.

```
        continue
    pt_3d,A=fun.direct_triangle(k1,rt1,k2,rt2,u,v,up,vp)#calculate 3D point by direct tringle method
    check_pt = np.dot(A, pt_3d)
    check_pt = [x**2 for x in check_pt]
    error_pt = sum(check_pt) ** 0.5
    if error_pt < 10:
        r, g, b = image[v, u][2], image[v, u][1], image[v, u][0]
        numm=[pt_3d[0],pt_3d[1],pt_3d[2],r,g,b]
        p3d.append(numm)
```