15CSE363 – Principles of Digital Image Processing

Sky horizon detector

M.SARATH CHANDRA-BL.EN.U4CSE17072

D.VENKATA RAJU-BL.EN.U4CSE17532

M.SAI NIKHIL-BL.EN.U4CSE17074

Problem Statement:

- An algorithm that can differentiate between sky pixels and other pixels to create a horizon that can be used for further image registration and processing.

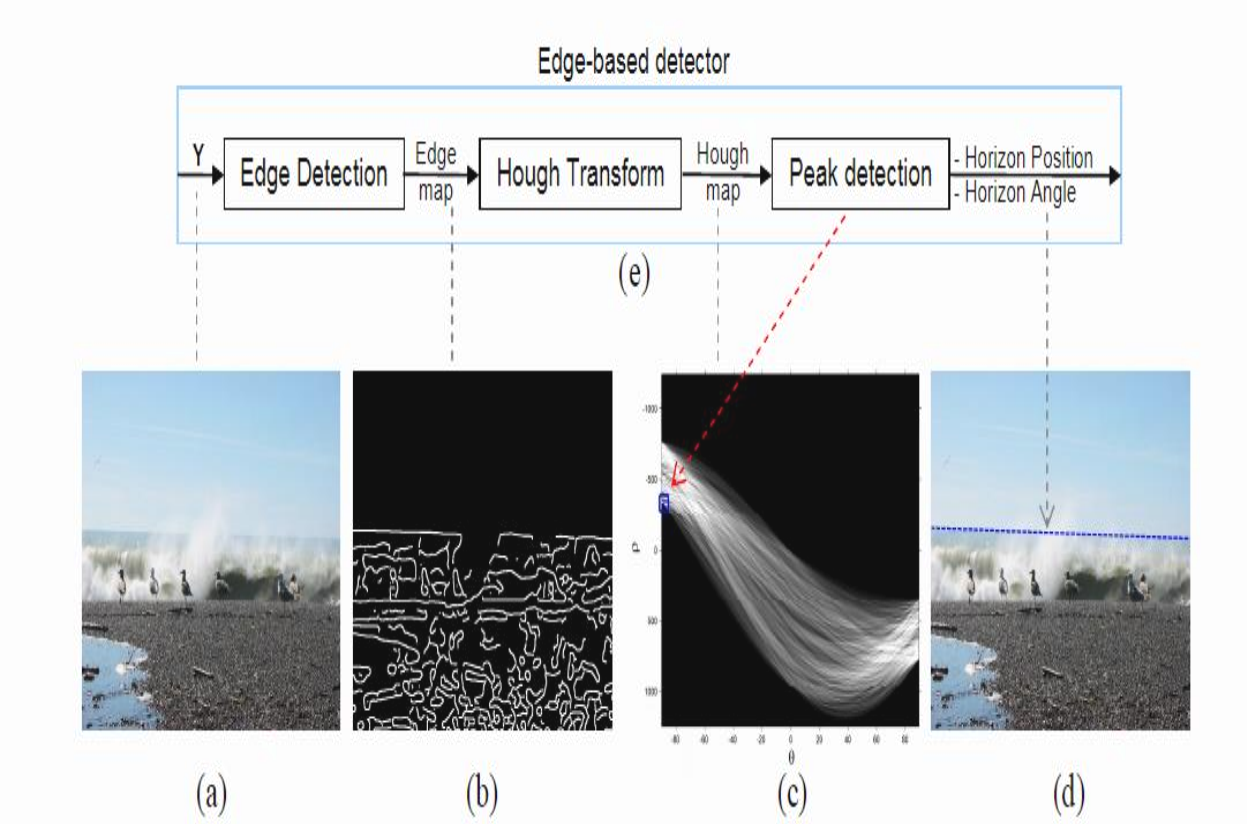
Motivation:

- Horizon detection in still images or video sequences contributes to the applications like image understanding, image registration and image quality enhancement. These results can be used to aid UAV and also be used in geology for the analysis of rock formation and layer detection.

Proposed Solution:

- Using an edge-based detector the principle edge which separates the sky from the rest of the image can be found. The detector uses the concepts of canny edge detector and Hough map transforms. Processing the image through a canny edge detector results in an edge image that contains the significant edges. Transforming the edge image using the Hough map transform straight lines can be found through which the principle edge line can be found.

WORKFLOW DIAGRAM:



Implementation:

The given image pixels are divided into two gaussian distributions sky and ground in RGB space and then a line segment which separates these two is found by maximizing the optimization criteria.

```
def msc(y,m,x,c):
  result = y-(m*x)-c
  return result
```

```
for m in range(len(slope)):
    for c in range(len(inter)):
      sky = []
      ground = []
      for i in range(xsize):
        for j in range(ysize):
          u = msc(j,slope[m],i,inter[c])
          v = (-1*inter[c])
          if(u*v>0):
            sky.append(himg[j][i])
          else:
            ground.append(himg[j][i])
```

By using the above code the sky and ground pixels are categorized based on the slopes and intercepts defined, based on the formula derived from the Hough line transform technique.

```
        co_sky = np.cov(sky)
        co_ground = np.cov(ground)
        co_sky_det = lg.det(co_sky)
        co_ground_det = lg.det(co_ground)
        co_sky_eig = lg.eig(co_sky)
        co_ground_eig = lg.eig(co_ground)
```

Finding the determinant and eigenvalues of the covariance matrices of sky and ground pixel arrays.

```
  r = 1/(co_sky_det + co_ground_det +
(co_sky_eig[0][0]+co_sky_eig[0][1]+co_sky_eig[0][2])**2 +
(co_ground_eig[0][0]+co_ground_eig[0][1]+co_ground_eig[0][2])**2)

        if (r > r_max):

          r_max = r

          maximum = [slope[m],inter[c]]

      except Exception:

        pass

  return maximum
```

Using the above values we define a optimization term ‘ r ‘ that is maximized and the values of slope and intercept of that particular iteration is returned.

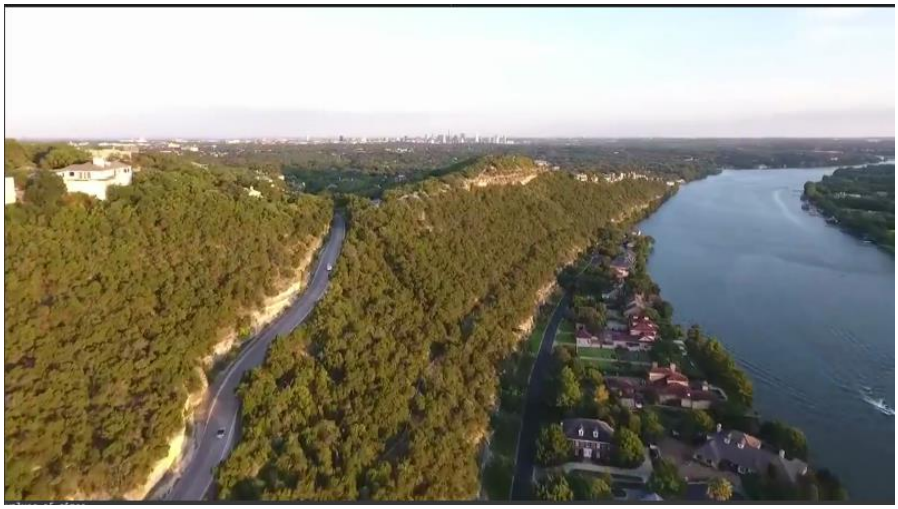
```
def drawline(himg , horizon):

  xsize = himg.shape[1]
```

```
print("xsize",xsize)

m = horizon[0]

b = horizon[1]

y2 = int(m*(xsize-1)+b)

cv2.line(himg ,(0,int(b)),(xsize-1 , y2),(0,0,255) , 2)

cv2_imshow(himg)
```

Using these returned values of slope and intercept a best fitting line is drawn on the input image and that image is the required output.

Result Analysis:

a. Input image



b. Output Image

FUTURE SCOPE:

The accuracy of the project can be increased by combining both the edge and color detection concepts together to make the project more reliable.

This can be integrated with the projects which provide visual assistance for the unmanned aerial vehicles and in image registration processes.

LITERATURE SURVEY:

https://www.researchgate.net/publication/253820856_Horizon_detection_based_on_sky-color_and_edge_features

http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV0405/NICKERSON/av.html

https://www.researchgate.net/publication/337742485_High-Speed_Horizon_Line_Detection_Algorithm_Using_Dual_Hough_Algorithm_for_Marine_Image

https://github.com/k29/horizon_detection