Course Project Report

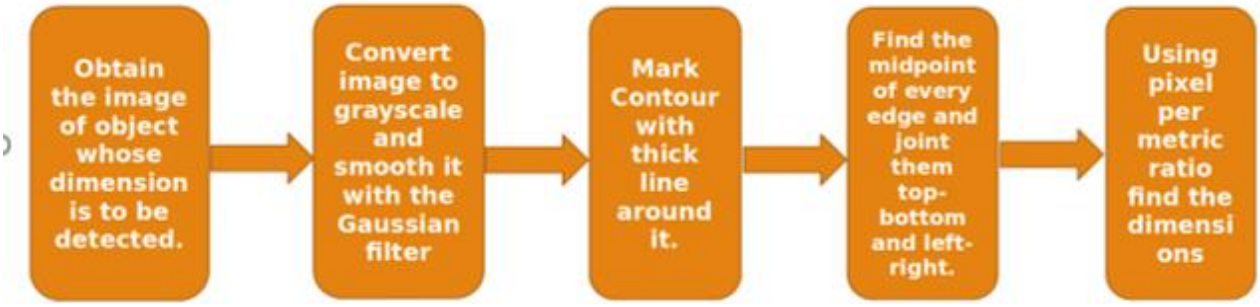
Title: Dimension Detection of Object using Computer Vision.

Presented by: 1) Saurabh Deshmukh-L9-161533

2) Kalpesh Mahajan-L24- 161671

3) Siddhi Chavan-L35-161076

Algorithm:



In the project, while measuring the dimensions of any object/shape the dimensions of coin are taken as reference in the camera frame. And all other objects dimension are measured by the reference of the dimensions of the coin. For, larger application other object can be taken as reference for measuring the dimensions of bigger objects which is to be initiated in the program. This code can work with any type of camera for real time dimensions detection and even with the Laptop's front camera.

Working: Take the filtered image and apply the contours on the image. After finding the contours we sort the contours such that the leftmost contours are the first. Sorting of the contours is done to recognize the reference object. The reference object is the object of known dimension. By drawing the contours, we measure the pixels covered by the object. And store the number of pixels covered by the object terms of its pixel height and Pixel width:

$$\alpha = \frac{d}{p}$$

Where α – distance per pixel.

d – Real dimension of reference object

p – distance in number of pixels.

It is necessary to have a reference object of known dimensions. And always try to keep reference object at he left. We keep the reference object in the left so that is considered as the first contour. By detecting the dimensions of this contours in pixels we find the ratio of actual size to the pixels covered be the contours. This ratio gives us size of the object per unit pixel. The same ratio is determined for the height and width of the reference image. This is the standard ratio used for calculating the dimension of any object in the frame by referring the reference object. From the contour of the test object we find the pixels covered by the test object. Thus, by multiplying the above ratio with the number of pixels we get the original dimension of the test object in the frame.

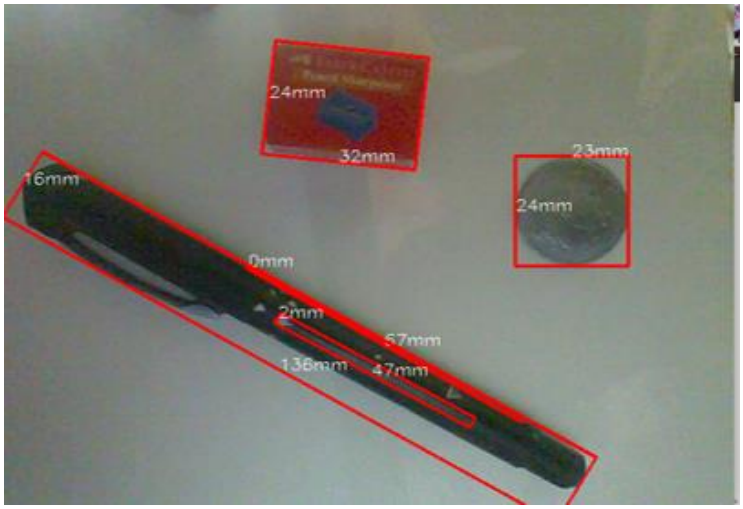
Code:

```
import cv2
import numpy as np
from scipy.spatial import distance
def fil_img(img):
    grey_img=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    grey_img_blur=cv2.GaussianBlur(grey_img,(5,5),0)
    canny=cv2.Canny(grey_img_blur,10,70)
    canny=cv2.dilate(canny, None, iterations=1)
    canny=cv2.erode(canny, None, iterations=1)
    return canny
def midpoint(A,B):
        return ((A[0]+B[0]) * 0.5, (A[1]+B[1]) * 0.5)
cap=cv2.VideoCapture(0)
while True:
    ret,frame=cap.read()
    edged=fil_img(frame)
    cnts,hierarchy=cv2.findContours(edged.copy(),
cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
    sorted_cnts = sorted(cnts, key=lambda cnt: cv2.minAreaRect(cnt)[0],reverse=True)
    data=[]
```

```
    for c in sorted_cnts:
        if cv2.contourArea(c)<100 :
            continue
        rect=cv2.minAreaRect(c)
        box = cv2.boxPoints(rect)
        box = np.array(box, dtype="int")
        cv2.drawContours(frame,[box],0,(0,0,255),2)
        #cv2.drawContours(frame,[c],-1,(0,255,0),3)
        data.append(box)
        tl, tr, br, bl=box
        xup,yup=midpoint(tl,tr)
        xdown,ydown=midpoint(bl,br)
        xl,yl=midpoint(tl,bl)
        xr,yr=midpoint(tr,br)
        ref_w=24/distance.euclidean((data[0][0]),(data[0][1]))
  #data[0][1][0]
        ref_h=24/distance.euclidean((data[0][0]),(data[0][3]))
#data[0][1][1]
        height=ref_w*distance.euclidean((xup,yup),(xdown,ydown))
        width=ref_h*distance.euclidean((xl,yl),(xr,yr))
cv2.putText(frame,str(int(width))+'mm',(int(xup),int(yup)),cv2.FONT_HERSHEY_SIMPLEX,0.
5,(255, 255, 255),1)
cv2.putText(frame,str(int(height))+'mm',(int(xr),int(yr)),cv2.FONT_HERSHEY_SIMPLEX,0.5,(
255, 255, 255),1)
    cv2.imshow('SORTED',frame)
    if cv2.waitKey(1)==13:
        break
cap.release()
cv2.destroyAllWindows()
```

Output:

Input Image



Output Image

Conclusion:

This method requires a single camera and helps us find the length and breadth of an object accurately with a negligible percentage of error. Hence, we propose a method which is reliable, easy to implement, cost effective and promises a minimal overhead in calculation.