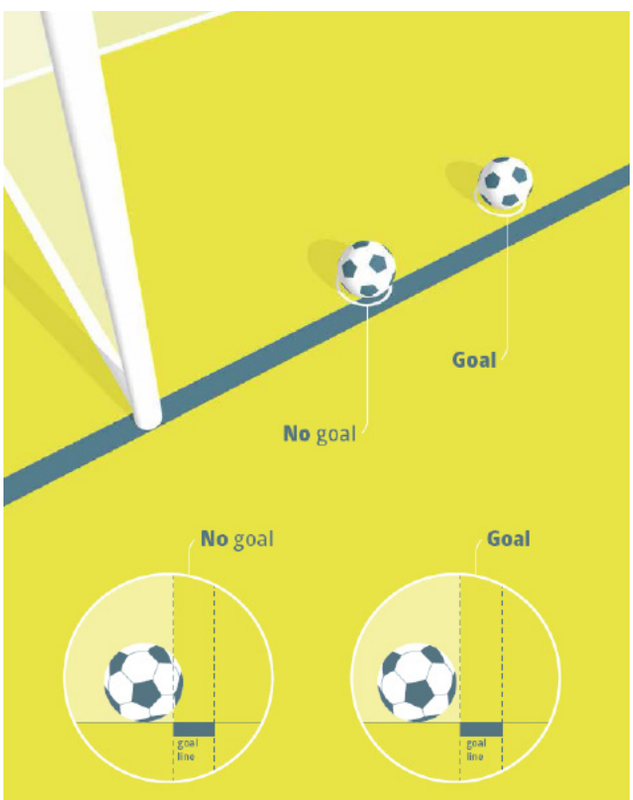
Goal Line Technology

Report
S N Bibhudutta
21EC30043

Table Of Contents

Introduction

The aim of this project is to build a system that can detect whether a goal has been scored or not by checking the postion of the ball with respect to the goal line.



Methods Included are:-

1. Thresholding: Converts the image into binary which removes unwanted data and helps in edge detection.

2. Morphological Transformation: It removes the white noise present in the converted binary image.

3. Contour Detection: Detects edges of the objects and is used to determine the shape of an object. There are a lot of parameters like position, perimeter and area can be found out using contours.

4.Movement Detection:- Detects the movement of an object by comparing the relative positions of contours w.r.t the previous frame.

5. Goal Detection:- When the ball crosses the line, it is detected as goal.

Thresholding

There are two types of thresholding:-

1. Simple
2. Adaptive

1. Simple

a) For every pixel, same threshold value is applied.
b) If the pixel value is smaller than the threshold, it is set to 0, otherwise it is set to a maximum value.
c) The fuction used is:- "opencv.threshold()"
d) There are three arguments of the this funcion:-
i) Grayscale Image (Avg. of Red, Blue and Green Pixels)
ii) Threshold Value
iii) Max value to be assigned to pixels which have greater value than threshold. (We used "cv.THRESH_BINARY")

2. Adaptive

a) Different threshold value for different pixels.
b) Algorithm determins threshold on the basis of a small region around it.
c) The function used is:- "opencv.adaptiveThreshold()"
d) Similar arguments as there in simple thresholding.

Morphological Transformation

There are two types of basic morphological transformation operators:-

1. Erosion
2. Dilation

1. Erosion

a) A pixel in the original image (either 1 or 0) will be considered 1 only if all the pixels under the kernel is 1, otherwise it is made to zero.
b) All the pixels near boundary will be eroded depending upon the size of kernel.
c) The thickness of the object decreases.
d) The fuction used is:- "opencv.erode()"
e) There are three arguments of this function:-
i) Input image
ii) kernel
iii) Number of times the kernel is moved.

2. Dilation

a) A pixel in the original image (either 1 or 0) will be considered 1 if any of the pixels under the kernel is 1, otherwise it is made to zero.
b) the thickness of the object increases.
c) The fuction used is:- "opencv.dilate()"
d) Same arguments as there in previous case

Contour Detection

a) A curve joining all the continuous points (along the boundary), having same color or intensity is called contour.
b) They are useful tool for shape analysis and object detection.
c) We use binary images to save data and for better accuracy.

1. Finding Contours

a) The funtion used is "opencv.findContours()".
b) It uses an algorithm (Topological Structure Analysis of Digitized Binary Images by Border Following).
c)There are three arguments of this function:-
i) Source Img.
ii) Contour Retrieval Mode ("opencv.RETR_TREE").
iii) Contour Approximation Method ("opencv.CHAIN_APPROX_SIMPLE").
d) Contour is a numpy array of (x,y) coordinates of of boundary points of the object.

A. Contour Approximation Method

a) "opencv.CHAIN_APPROX_NONE":- Stores all counter points
b) "opencv.CHAIN_APPROX_SIMPLE":- Compresses horizontal, vertical or diagonal neighbours and leaves their end points to save memory.

2. Drawing Contours

a) The funtion used is "opencv.drawContours()".
b) It draws contour outlines if thickness > 0 otherwise fills the area bounded.
c) The arguments of this function are:-
i) Source Img
ii) Contours pass in list format
iii) Index of contours for individual contour drawing (To draw all contours, pass - 1)

3. Contour Features

a) Moments:- Used to calculate features like centroid, area, etc. Function used is "opencv.moments()".
i) Centroid

```
cx = int(M['m10']/M['m00'])
cy = int(M['m01']/M['m00'])
```

b) Area:- Funtion used "opencv.counterArea()".
c) Perimeter:- Function used "opencv.arcLength(cnt, True) {The 'True' specifies whether the shape is closed or not}.
d) Approximation:- Approximates a contour shape to another shape with less number of vertices.
Function used "opencv.PolyDP(cnt,epsilon,True) {The 'Epsilon' is the maximum distance from contour to approximated contour. 1-5% of the original contour length. This is reason we need perimeter of the contour line}.

Shape Detection

a) After calculating the number of vertices by using the function "opencv.PolyDP(cnt,epsilon,True), we use a 'if-else' statement to specify the shape of the object.
b) If, the number of vertices are 3 then it's a triangle, if the number of vertices is 4 then it's a square or rectangle,
c) If, the number of vertices are 5 or more, then it is considered as a circle.

Movement Detection

a) The relative distance between the centroid of two contours are calculated.
b) If the relative distance travelled is greater than a threshold value, then the body is said have some considerable movement.

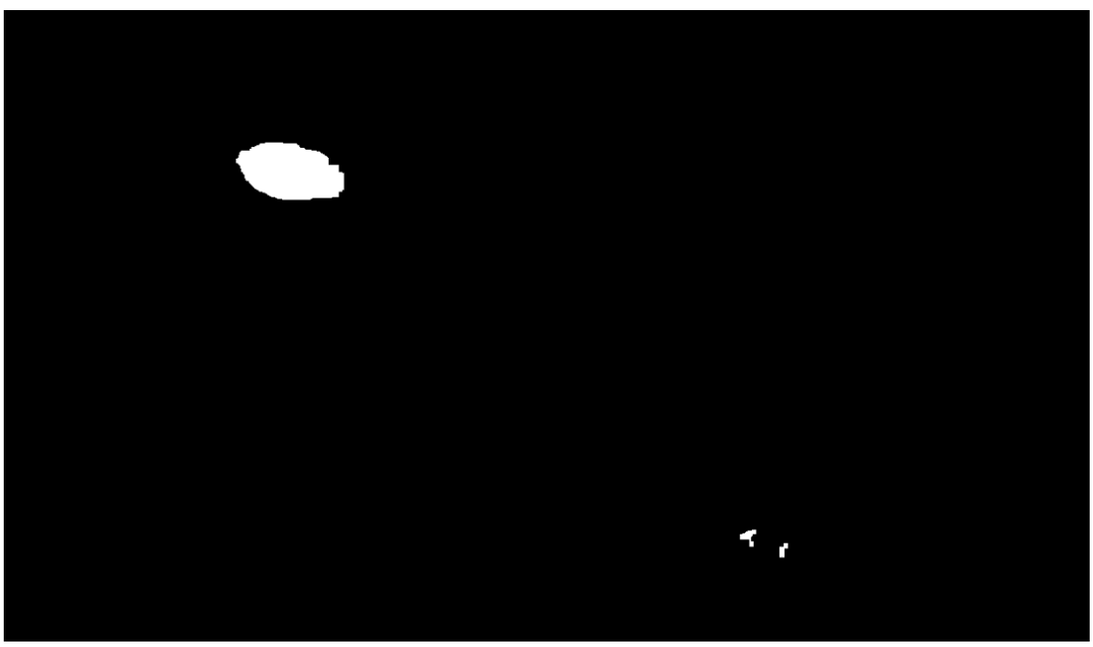
```
def is_nearby_in_array(x, y):
    is_nearby = False

    for (x0, y0) in object_positions:
        if math.sqrt((x - x0)**2 + (y - y0)**2) <= 50:
            is_nearby = True
            break

    return is_nearby
```

Images From The Model

Ball In The Frame

Thresholding

Images From The Model



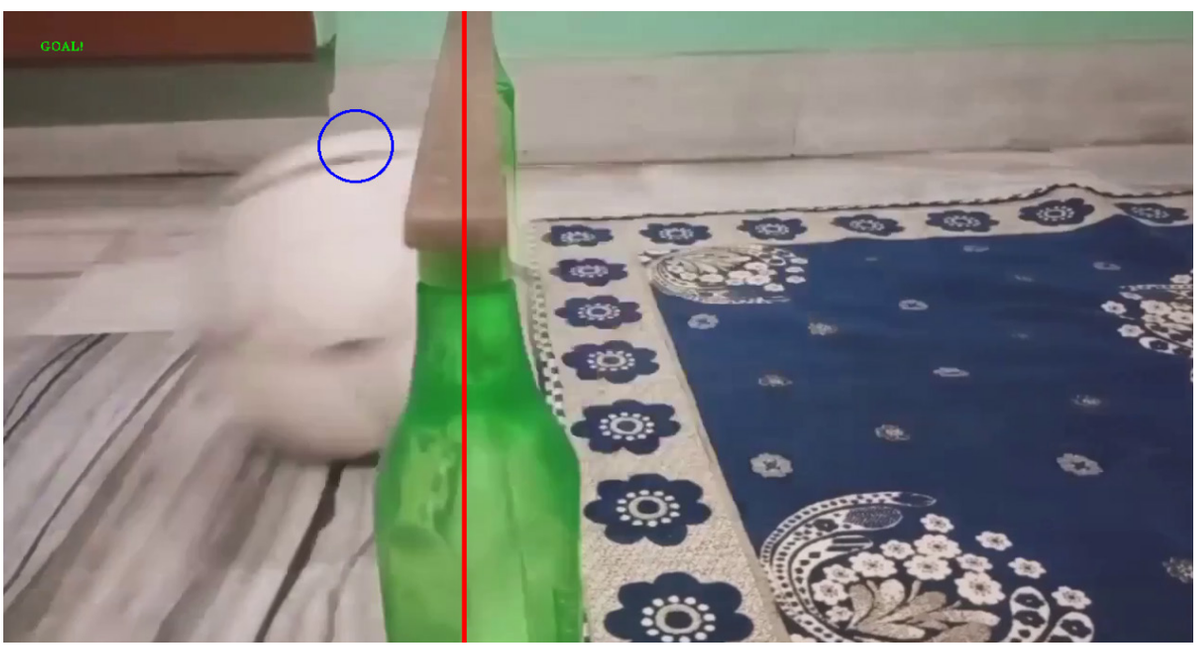
Denoising

Contour Detection

Goal Detection

```python
# Overlay a goal line on the image
goal_line_x = int(frame_width * GOAL_LINE_LOCATION / 100)
opencv.line(
    original,
    (goal_line_x, 0),
    (goal_line_x, frame_height),
    GOAL_LINE_COLOR,
    GOAL_LINE_THICKNESS
)
```

```python
# goal is scored if the contour is inside the goal line
if x < goal_line_x:
    opencv.putText(original, "GOAL!", (50, 50), opencv.FONT_HERSHEY_COMPLEX_SMALL, .7, (0, 255, 0))
    print(f"Goal found in frame number: {frame_number}")
```



Thank You For Reading