Champhunt Cricket Ball Detection Documentation

- Jiyanshu Dhaka

I wrote Python script to detect cricket ball in video. I used OpenCV and NumPy for this purpose. script reads video frame by frame. It then processes each frame to detect red-colored objects. Finally, it applies checks to ensure that only cricket ball is detected. Below, I explain entire script in detail.

Libraries

First, I import necessary libraries:

```
import cv2
import numpy as np
import os
```

- `cv2`: Used for video processing and object detection.
- `numpy`: Helps with mathematical operations.
- `os`: Used for file handling (though not extensively in this script).

Input and Output Paths

Next, I define paths for input and output videos:

```
video_path = '/content/in2.mp4'
output_video_path = '/content/processed_video.avi'
```

- input video is located at `/content/in2.mp4`.
- processed video will be saved as `/content/processed_video.avi`.

Video Loading

I load video using `cv2.VideoCapture`:

```
cap = cv2.VideoCapture(video_path)
```

- Frames are read sequentially.
- This ensures smooth processing of video.

Video Properties

I extract properties of video:

```
fps = cap.get(cv2.CAP_PROP_FPS)
frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
```

- `fps`: Retrieves number of frames per second.
- `frame_width` and `frame_height`: Provide video dimensions.

VideoWriter Object

I create `VideoWriter` object to save processed video:

```
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter(output_video_path, fourcc, fps, (frame_width, frame_height))
```

- `fourcc`: Defines codec. I used `XVID`.
- `out` object writes processed frames into output file.

Ball Detection Function

I define function to filter out non-ball objects:

```
def is_ball(contour):
    area = cv2.contourArea(contour)
    perimeter = cv2.arcLength(contour, True)
    if perimeter == 0:
        return False
    circularity = 4 * np.pi * (area / (perimeter * perimeter))
    return 150 < area < 800 and 0.85 < circularity < 1.15
```

- `area`: Calculates size of object.
- `perimeter`: Measures object's boundary length.
- Circularity formula:

$$\text{Circularity} = \frac{4\pi \times \text{Area}}{\text{Perimeter}^2}$$

- Thresholds ensure object is circular and within valid size range.

Frame Processing

I process each frame in loop:

```
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break
```

- Each frame is read using `cap.read()`.
- loop exits when no frames are left to process.

Color Conversion

I convert frame to HSV color space:

```
hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
```

- HSV is preferred for color-based segmentation over BGR.

Red Color Masks

I create masks to detect red regions:

```
lower_red1 = np.array([0, 120, 70])
upper_red1 = np.array([10, 255, 255])
lower_red2 = np.array([170, 120, 70])
upper_red2 = np.array([180, 255, 255])

mask1 = cv2.inRange(hsv, lower_red1, upper_red1)
mask2 = cv2.inRange(hsv, lower_red2, upper_red2)
red_mask = mask1 | mask2
```

- Two ranges cover light and dark red shades.
- masks are combined using bitwise OR operator.

Blur and Contours

I reduce noise and find contours:

```
blurred = cv2.GaussianBlur(red_mask, (7, 7), 0)
contours, _ = cv2.findContours(blurred, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

- Blurring smoothens mask to reduce noise.
- Contours mark boundaries of detected objects.

Ball Validation

I check each contour:

```
for contour in contours:
    if is_ball(contour):
        (x, y), radius = cv2.minEnclosingCircle(contour)
        center = (int(x), int(y))
        radius = int(radius)
        cv2.circle(frame, center, radius, (0, 255, 0), 2)
        cv2.putText(frame, "Cricket Ball", (int(x) - 10, int(y) - 10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)
        break
```

- `is_ball` function filters non-ball objects.
- I draw green circle around detected ball and label it.

Save and Release

Finally, I save each frame and release resources:

```
out.write(frame)
cap.release()
out.release()
```

- Processed frames are written to output file.
- All resources are freed after processing.

so i detect cricket ball in video using OpenCV. It combines color segmentation, contour filtering, and temporal consistency. processed video is saved as `processed_video.avi`.