**CODE**

import numpy as np

import cv2

import matplotlib.pyplot as plt

def colorConvert(image):

  return(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))

cap = cv2.VideoCapture('/content/video\_1.mp4')

#Randomly selecting 30 frames

frame\_get = cap.get(cv2.CAP\_PROP\_FRAME\_COUNT) \* np.random.uniform(size = 30)

#Storing captured frames in an array

frames = []

for i in frame\_get:

  cap.set(cv2.CAP\_PROP\_POS\_FRAMES, i)

  ret, frame = cap.read()

  frames.append(frame)

cap.release()

frame\_median = np.median(frames, axis = 0).astype(dtype = np.uint8)

plt.imshow(colorConvert(frame\_median))

frame\_avg = np.average(frames, axis = 0).astype(dtype = np.uint8)

plt.imshow(colorConvert(frame\_avg))

frame\_sample = frames[0]

plt.imshow(colorConvert(frame\_sample))

gray\_frame\_median = cv2.cvtColor(frame\_median, cv2.COLOR\_BGR2GRAY)

plt.imshow(colorConvert(gray\_frame\_median))

gray\_frame\_sample = cv2.cvtColor(frame\_sample, cv2.COLOR\_BGR2GRAY)

plt.imshow(colorConvert(gray\_frame\_sample))

bg\_removed\_frame = cv2.absdiff(gray\_frame\_sample, gray\_frame\_median)

plt.imshow(colorConvert(bg\_removed\_frame))

frame\_blur = cv2.GaussianBlur(bg\_removed\_frame, (11,11), 0)

plt.imshow(colorConvert(frame\_blur))

ret, frame\_threshold = cv2.threshold(frame\_blur, 0, 255, cv2.THRESH\_BINARY + cv2.THRESH\_OTSU)

plt.imshow(colorConvert(frame\_threshold))

(contours, \_ ) = cv2.findContours(frame\_threshold.copy(), cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

for i in contours:

  x, y, width, height = cv2.boundingRect(i)

  cv2.rectangle(frame\_sample, (x,y), (x + width, y + height), (123,0,255), 2)

plt.imshow(colorConvert(frame\_sample))

frame\_count = 0

while (frame\_count < frame\_tot - 1):

  frame\_count+=1

  ret, frame = cap.read()

  # Converting frame to grayscale

  gray\_frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

  # Calculating Absolute Difference between Current Frame and Median Frame

  dframe = cv2.absdiff(gray\_frame, gray\_frame\_median)

  # Applying Gaussian Blur to reduce noise

  blur\_frame = cv2.GaussianBlur(dframe, (11,11), 0)

  # Binarizing frame - Thresholding

  ret, threshold\_frame = cv2.threshold(blur\_frame, 0, 255, cv2.THRESH\_BINARY + cv2.THRESH\_OTSU)

  # Identifying Contours

  (contours, \_ ) = cv2.findContours(threshold\_frame.copy(), cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

  # Drawing Boundary Boxes for each Contour

  for i in contours:

    x, y, width, height = cv2.boundingRect(i)

    cv2.rectangle(frame, (x,y), (x + width, y + height), (123,0,255), 2)

    video\_writer.write(cv2.resize(frame, (640,480)))

# Releasing Video Object

cap.release()

video\_writer.release()

OUTPUT

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |