Curneu MedTech Innovations Private Limited - Internship Task

Pixel Variation Detection in video

USING ML ALGORITHMS

QUESTION

Understand the given video clip and Determine the Pixel Variations across the Video using Machine Learning Algorithms in C++ from scratch. Represent the Variations in Graph with RGB values and interpret your results.

VIDEO DESCRIPTION:

- The given video is in the mp4 format of length 02:42 minutes
- The frame size is 426x240.

- It contains 25.00 frames/second

PROBLEM STATEMENT:

- The given video is to be converted to frames and pixels are to be extracted.
- The variations in RGB color format of the extracted pixels should be displayed in a graphical format using C++.

PROBLEM APPROACH:

1. The pixels are extracted from each frame using OpenCV in C++ and appended in an array (3 arrays for each of the RGB color codes).
2. The array is then appended into a CSV file containing attributes R,G,B standing for the RGB values of the current frame respectively.
3. In addition to this the time stamp can also be added to the CSV file to arrive at better results.

IMPLEMENTATION:

#VIDEO - FRAME CONVERSION

I have broken down the video into frames using python OpenCV

```
import cv2
vidcap = cv2.VideoCapture('Desktop/DS.mp4')
success,image = vidcap.read()
count = 0
#Get frame height and width to access pixels
height, width, channels = image.shape
r = []
g = []
b = []
```

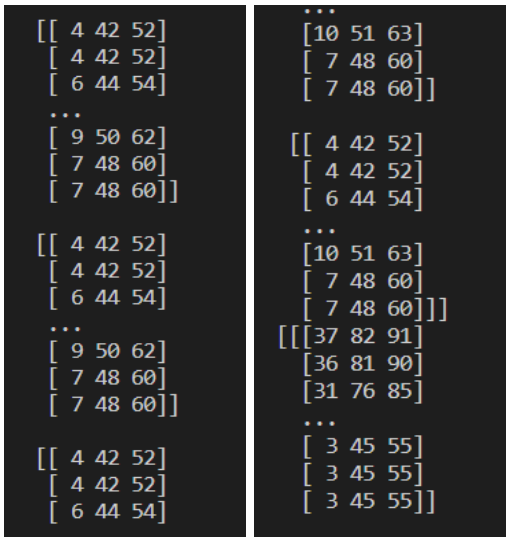
```
#Accessing BGR pixel values

for x in range(0, width) :

     for y in range(0, height) :

          b.append(image[x,y,0]) #B Channel Value

          g.append(image[x,y,1]) #G Channel Value

          r.append(image[x,y,2]) #R Channel Value
```

This returns the RGB values of the current frame. This is appended to a CSV file

Next, these RGC codes are processed using C++ by reading the CSV file

```
public int getRGB(int index) {
int[] p = getPattern(index);
return getElement(p[0]) << 16 | getElement(p[1]) << 8 |
getElement(p[2]);
}
```

OUTPUT:



Draw patterns of the codes using the following lines of code

```
public int[] getPattern(int index) {
    int n = (int)Math.cbrt(index);
    index -= (n*n*n);
    int[] p = new int[3];
    Arrays.fill(p,n);
    if (index == 0) {
        return p;
```

```
            }
            index--;
            int v = index % 3;
            index = index / 3;
            if (index < n) {
                p[v] = index % n;
                return p;
            }
            index -= n;
            p[v      ] = index / n;
            p[++v % 3] = index % n;
            return p;
        }


int main(int argc, char const *argv[])

{

  std::string video_file;

  // Read video file

  if(argc > 1)

  {

    video_file = argv[1];

  } else

  {

    video_file = "C:\Users\Mahima\Downloads\Question 3\Question
3\DS.mp4";

  }



  VideoCapture cap(video_file);

  if(!cap.isOpened())

    cerr << "Error opening video file\n";



  // Randomly select 25 frames
```

```
  default_random_engine generator;
  uniform_int_distribution<int>distribution(0,
  cap.get(CAP_PROP_FRAME_COUNT));

  vector<Mat> frames;
  Mat frame;

  for(int i=0; i<25; i++)
  {
    int fid = distribution(generator);
    cap.set(CAP_PROP_POS_FRAMES, fid);
    Mat frame;
    cap >> frame;
    if(frame.empty())
      continue;
    frames.push_back(frame);
  }
  // Calculate the median along the time axis
  Mat medianFrame = compute_median(frames);

  // Display median frame
  imshow("frame", medianFrame);
  waitKey(0);
}
#Color list generator
  function getColorList(8) {
```

```
    let retval =
['rgb(213,62,79)','rgb(244,109,67)','rgb(253,174,97)','rgb(254,224,139)'
,'rgb(230,245,152)','rgb(171,221,164)','rgb(102,194,165)','rgb(50,136,18
9)'];

    return retval;

  }
```

CHALLENGES:

Frame conversion and color code extraction consumed a lot of time.

It was difficult to extract RGB codes for each and every frame and handle missing values.

Missing values had to be solved by using color code technique.

FUTURE WORK:

The frame conversion from the video can be implemented in C++ and pixels can be generated in the said language.

CONCLUSION:

Variations in Graph with RGB values were derived and it was found that the pixel i.e clarity is increasing as the video plays. Towards the end high pixel has been generated.