CS 6327 Video Analytics Assignment 2

Instructor: Dr. B. Prabhakaran TA: Kanchan Bahirat

Due - Feb 13th, 2016

The only submission on eLearning is accepted.

Submitted by :

Himanshu Kandwal

Net id : hxk154230

Description:

For the first part of the assignment, capture a live video from the webcam in which you are moving an orange(or orange colored ball) in front of the webcam. (Note that this is not an offline captured video but a live video captured using the webcam). For each captured image frame in a live video, perform the following tasks:

1. Convert it to HSV domain.
2. Detect the orange in the frame using color-based object detection used in the previous assignment. Convert the orange into blue colored orange and mark it with a bounding box.

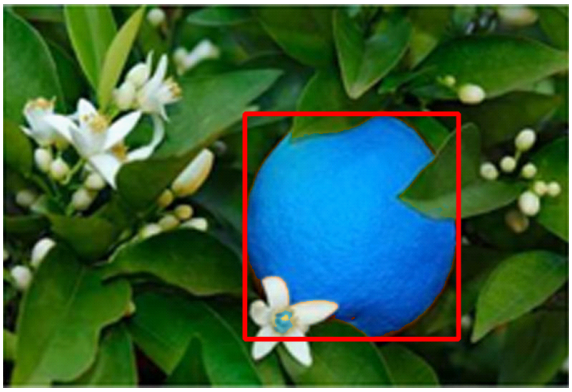
For the second part of the assignment, write a program to measure the diameter of an orange. (***Hint***: You can keep a reference object with the known dimensions in the scene and use that for determining the diameter of the orange.)

Desired Output:

For the first part of the assignment:

One of the frame of the input video:

The corresponding frame of the output video may look like:



So you will have two windows showing videos. One window will show the live video captured using the webcam and the second window will show the same video but with the color of orange changed to blue and orange is being marked with the bounding box.

For the second part of the assignment:

In a live video, you can put a reference object along with the orange. You can compute the diameter of the orange and display it after every 10 frames. (Note that in this case, estimated diameter of the orange may vary a little bit per computation)

Where to submit the assignment:

eLearning.

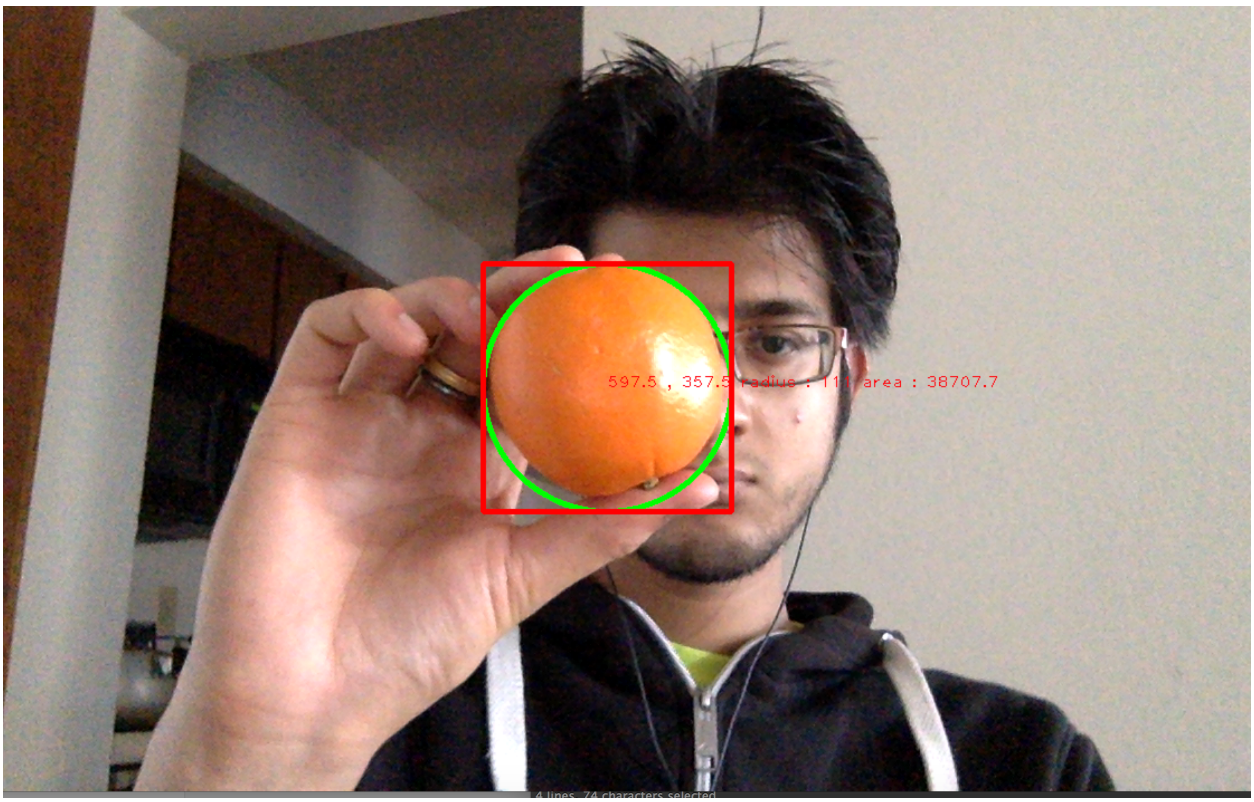
Late Submissions: Accepted. However, there will be a penalty when you are late.

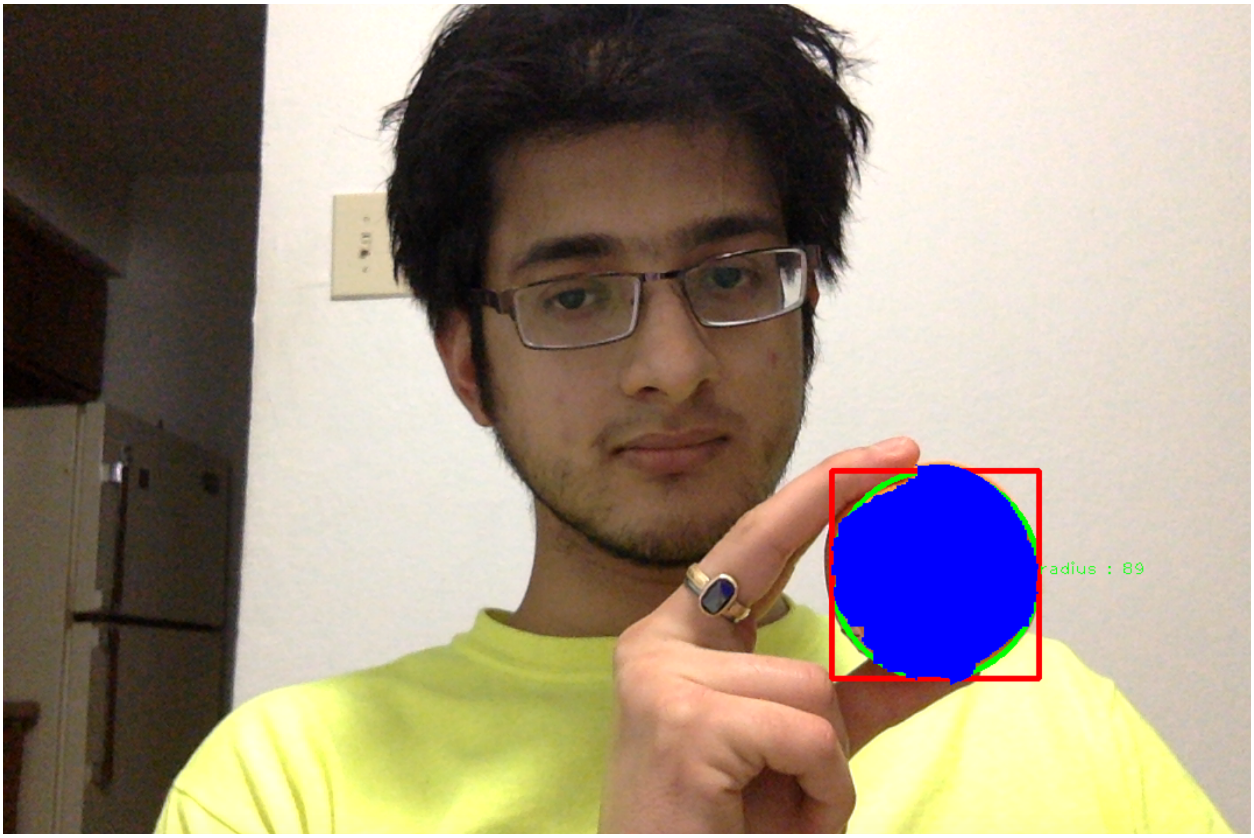
Rubrics:

Capture live video from webcam	– 20 points.
Color-based object detection	-- 10 points
Turning the color of orange to blue	-- 20 points
Marking the bounding box around the orange	-- 20 points
Performing mentioned tasks for the entire live video	-- 10 points

Determining the diameter of orange in centimeters -- 20 points

Solution :





Source code :

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include <opencv/cv.h>
#include <iostream>
#include <stdio.h>
#include <cstdlib>

using namespace cv;
using namespace std;

// Scalar ORANGE_MIN = Scalar(5, 50, 50);
// Scalar ORANGE_MAX = Scalar(18, 255, 255);

Scalar ORANGE_MIN = Scalar(7, 50, 50);
Scalar ORANGE_MAX = Scalar(22, 255, 255);

void morphOps(Mat &thresh){

   //create structuring element that will be used to "dilate" and "erode" image.
   //the element chosen here is a 3px by 3px rectangle
   Mat erodeElement = getStructuringElement(MORPH_RECT, Size(3,3));
   //dilate with larger element so make sure object is nicely visible
   Mat dilateElement = getStructuringElement(MORPH_RECT, Size(5,5));

   erode(thresh,thresh,erodeElement);
   erode(thresh,thresh,erodeElement);
   erode(thresh,thresh,erodeElement);
   erode(thresh,thresh,erodeElement);
}

int performColoring = 0;

int main(int argc, char** argv)
{
   int dp=1 , mindist=100, param1=300, param2=60, minradius=0, maxradius=0;
   IplImage* frame;

   performColoring = atoi(argv[1]);

   CvCapture* capture = cvCaptureFromCAM(0);

   if (!capture) {
      printf("Capture failure\n");
```

```
    return -1;
  }

  while (true)
  {
    frame = cvQueryFrame(capture);

    if (!frame)
      break;

    Mat imgOriginal(frame);

    Mat orangeDetectedImage, imgHSV;
    cvtColor(imgOriginal, imgHSV, COLOR_BGR2HSV);

      // // remove skin tone
      // Mat alpha, dst;
      // inRange(imgHSV, Scalar(0, 40, 60), Scalar(20, 150, 255), alpha);
      // bitwise_not(alpha,alpha);


      // //split source
      // Mat bgr[3];
      // split(imgHSV, bgr);

      // //Merge to final image including alpha
      // Mat tmp[4] = {bgr[0],bgr[1],bgr[2], alpha};
      // merge(tmp, 4, imgHSV);


    inRange(imgHSV, ORANGE_MIN, ORANGE_MAX, orangeDetectedImage);
    morphOps(orangeDetectedImage);

    vector< vector<Point> > contours;
    vector<Vec4i> hierarchy;
    vector<Point> approx;

    findContours(orangeDetectedImage.clone(), contours, hierarchy, RETR_EXTERNAL,
CV_CHAIN_APPROX_SIMPLE, Point(0, 0));

    for (int i = 0; i < contours.size(); i++)
    {
      approxPolyDP(cv::Mat(contours[i]),  approx,  cv::arcLength(cv::Mat(contours[i]), true) * 0.02,
true);

      if (std::fabs(cv::contourArea(contours[i])) < 100 || !cv::isContourConvex(approx))
```

```
            continue;

        cout << "shape detected !" << approx.size() << endl;

        // Detect and label circles
        if (approx.size() >= 5)
        {
            // Detect and label circles
            double area = cv::contourArea(contours[i]);
            cv::Rect r = cv::boundingRect(contours[i]);
            int radius = r.width/2;

            if (std::abs(1 - ((double)r.width / r.height)) <= 0.2 &&  std::abs(1 - (area / (CV_PI *
std::pow(radius, 2)))) <= 0.2) {
                cout << "circle detected !" << endl;
                int xpos = r.x + (r.width/ 2);
                int ypos = r.y + (r.height/ 2);

                Point center = Point(xpos, ypos);

                ostringstream convert;
                convert << xpos << " , " << ypos << " radius : " << radius;

                circle(imgOriginal, center, radius, Scalar(0, 255, 0), 3, 8, 0 );
                rectangle(imgOriginal, center + Point(-radius, -radius), center + Point(radius, radius),
Scalar(0, 0, 255), 3);
                putText(imgOriginal, convert.str().c_str(), center , 1, 1, Scalar(0,255,0));

                if (performColoring == 1) {
                    drawContours(imgOriginal, contours, i, Scalar(255, 0, 0), CV_FILLED, 8, hierarchy);
                }
            }

        }

    }

    imshow("Finding Orange : orangeDetectedImage", orangeDetectedImage);
    imshow("Finding Orange : imgOriginal", imgOriginal);
    //imshow("Finding Orange : imgHSV", imgHSV);

    int c=cvWaitKey(1);

    if (char(c) == 27)
        break;
```

```
    }

    return 0;
}
```