**BÁO CÁO BÀI TẬP 4**

**Nhóm 18**

1. Lương Hữu Phú Lộc – 1511844
2. Phạm Ngọc Khôi Nguyên – 1512221
3. Nguyễn Trọng Phúc – 1512534
4. Mai Thiện Quang – 1512640
5. **Yêu cầu**

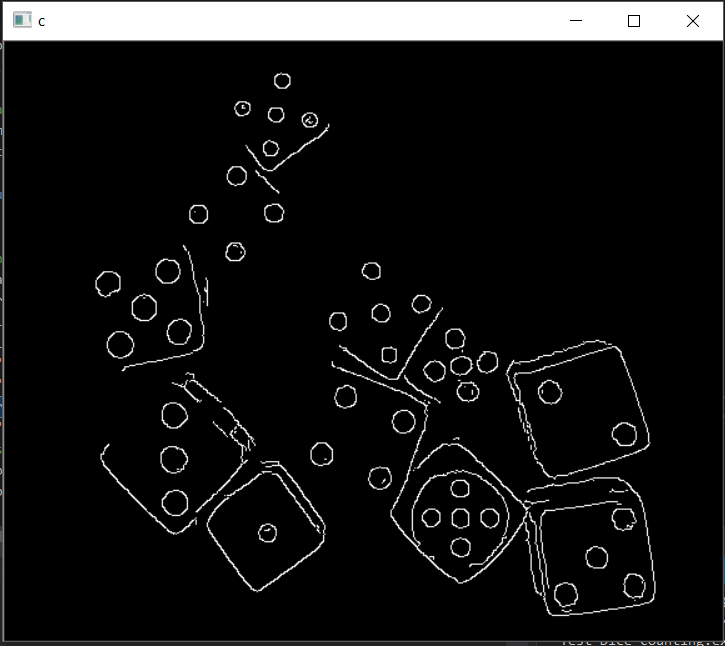
Cho hình các hột xí ngầu, đếm số chấm trên mặt xí ngầu dùng Matlab và C, in số bên cạnh hột.

Viết lại chương trình thay ảnh bằng video.

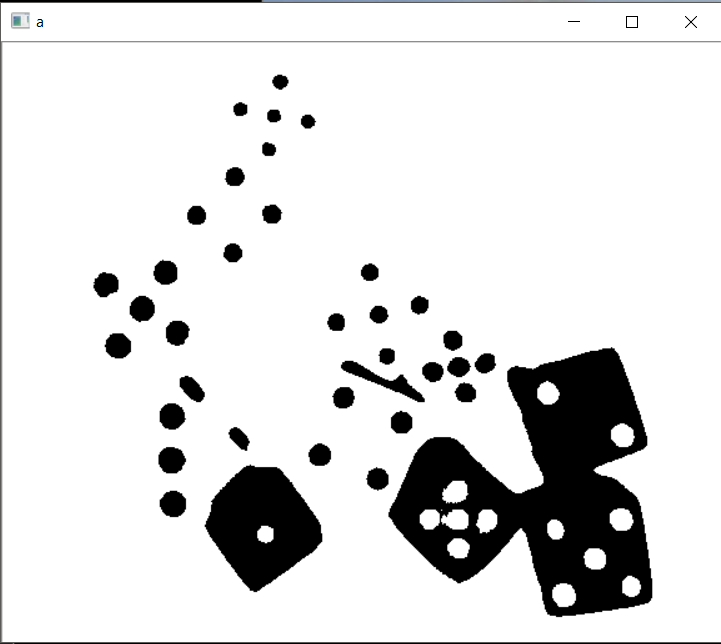


1. **Chương trình Visual C++**
2. **Nhận diện ảnh:**

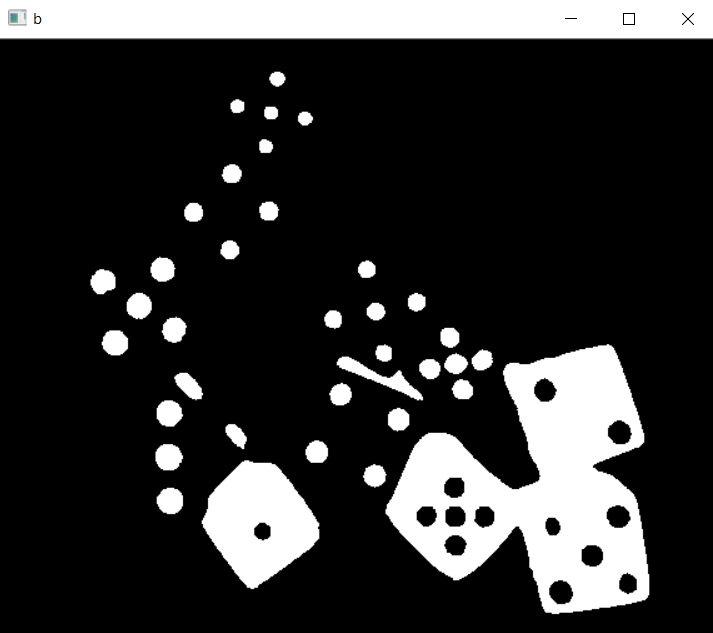
* Qua tìm hiểu, nhóm nhận thấy phương pháp thường sử dụng để nhận diện xí ngầu là đổi ảnh sang nhị phân sau đó tìm contours, dựa vào số contours để đếm số chấm.
* Tuy nhiên, do hình trên tương đối mờ, trong đó có xí ngầu với màu tương đối giống với màu nền (background) nên việc lấy contour hay cạnh một cách chính xác hình dạng của các con xí ngầu là tương đối khó khăn.
* Ví dụ lấy Canny của ảnh, sau khi đã lọc nhiễu, việc lấy contour của để lấy hình dạng các xí ngầu tương đối khó vì có các contour biên xí ngầu không tạo thành 1 hình dạng cụ thể.



* Vì vậy, nhóm sử dụng phương pháp khác là BlobDetect, là phương pháp tìm những điểm có tính chất giống nhau trong hình, cụ thể là màu sắc. Sử dụng BlobDetect, ta có thể tìm ra nhóm các điểm có màu đen, hay chấm đen trong hình.
* Vì hình trên gồm 2 loại xí ngầu là chấm đen và chấm trắng, nên nhóm tách làm 2 phần:
* Đầu tiên là đổi ảnh ban đầu sang nhị phân để tìm các xí ngầu có chấm đen trước bằng CV\_THRESH\_BINARY.



* Sau đó đổi ảnh sang nhị phân với CV\_THRESH\_INV để đảo ngược bit, như vậy các chấm trắng sẽ thành màu đen, và có thể nhận diện với BlobDetect.



* Sau khi tìm được các chấm đen, sử dụng lệnh kmean để phân nhóm các chấm này, các chấm trên cùng một xí ngầu sẽ thành 1 nhóm, cuối cùng ta chỉ cần đếm số chấm trong mỗi nhóm để biết được số chấm trên xí ngầu.
* Tuy nhiên, cách làm này có hạn chế: đó là ta phải biết được số lượng xí ngầu có trong hình, nếu không, việc phân nhóm sẽ bị sai. Đây cũng là hạn chế nhóm chưa khắc phục được.
* Kết quả sau khi thực hiện:



* **Chương trình thực hiện**

#include <opencv2/highgui/highgui.hpp>

#include <iostream>

#include <opencv2/opencv.hpp>

#include <stdio.h>

#include <stdlib.h>

using namespace cv;

using namespace std;

// Chuong trinh con tim so cham trong hinh,

// phan loai, dem so cham, sau do ghi len hinh

void dem(Mat input, Mat out, int num)

{

// Thiet lap cac thong so cho SimpleBlobDetector

SimpleBlobDetector::Params parameter;

// Area filtering to avoid small circles other than dice dots

parameter.filterByArea = true;

parameter.minArea = 100;

// Circularity filtering

parameter.filterByCircularity = true;

parameter.minCircularity = float(0.8);

vector<KeyPoint> keypoints; // To store circle blobs

// Set up detector with parameters

Ptr<SimpleBlobDetector>detector = SimpleBlobDetector::create(parameter);

detector->detect(input, keypoints); // Detect blobs

// Draw detected Circles as Green circles and use putText to

// display the total number of dots in the image

drawKeypoints(out, keypoints, out, Scalar(0, 255,0), DrawMatchesFlags::DRAW\_RICH\_KEYPOINTS);

vector <Point2f> points;

KeyPoint::convert(keypoints, points);

Mat labels, center;

char Text[20];

// Dung phuong phap kmean phan chia cac diem moi tim duoc

kmeans(points, num, labels, TermCriteria(CV\_TERMCRIT\_EPS + CV\_TERMCRIT\_ITER, 10, 1.0), 3, KMEANS\_PP\_CENTERS, center);

int i = 0;

int j = 0;

for (i = 0; i < num;i++)

{

Point2f a;

a.x = center.at<float>(i, 0);

a.y = center.at<float>(i, 1);

int temp = 0;

for (j = 0; j < labels.rows; j++)

{

if (labels.at<int>(j, 0) == i)

{

temp++;

}

}

sprintf\_s(Text, "%d", temp);

putText(out, Text, a, FONT\_HERSHEY\_SIMPLEX, 1.2, cvScalar(0, 0, 255), 3, CV\_AA);

}

}

/\*\* @function main \*/

int main()

{

Mat src, src\_fil;

Mat temp1,temp2;

Mat dst;

// Doc anh

src = imread("bai4.png", IMREAD\_COLOR);

if (!src.data)

{

return -1;

}

// Loc nhieu, chuyen sang anh nhi phan

bilateralFilter(src, src\_fil, 15, 80, 80);

cvtColor(src\_fil,src\_fil, CV\_BGR2GRAY);

threshold(src\_fil,temp1,110,255,CV\_THRESH\_BINARY);

threshold(src\_fil, temp2, 113, 255, CV\_THRESH\_BINARY\_INV);

imshow("a", temp1);

imshow("b", temp2);

Canny(src\_fil, dst, 20, 60, 3, false);

imshow("c", dst);

// Dem so cham dua vao so xi ngau, ghi len anh

dem(temp1, src, 7);

dem(temp2, src, 4);

// Hien thi ket qua

imwrite("result.png", src);

imshow("Output Image", src);

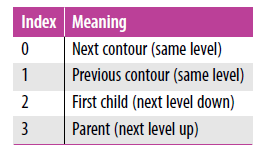
waitKey(0);

return 0;

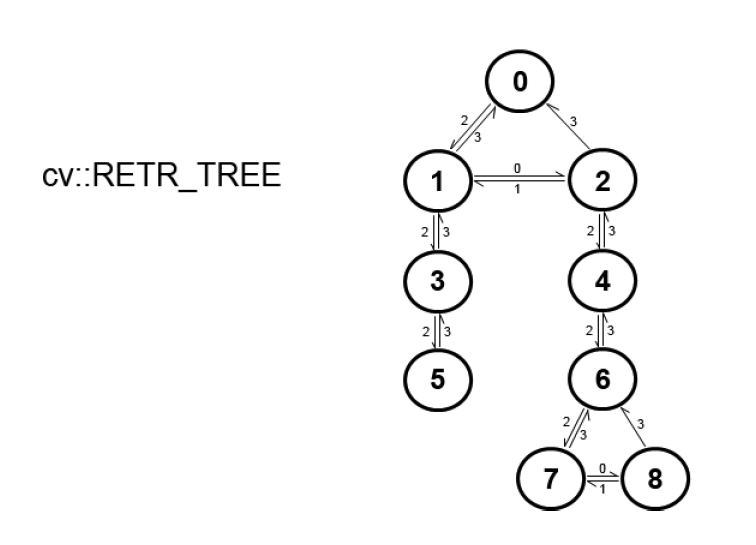
}

1. **Nhận diện video**

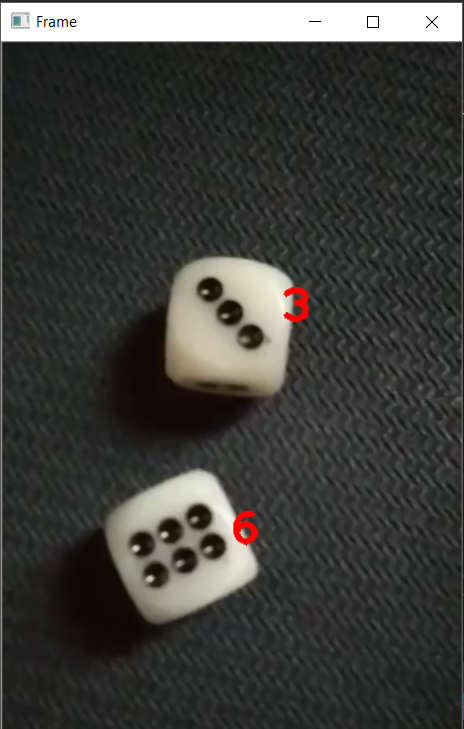
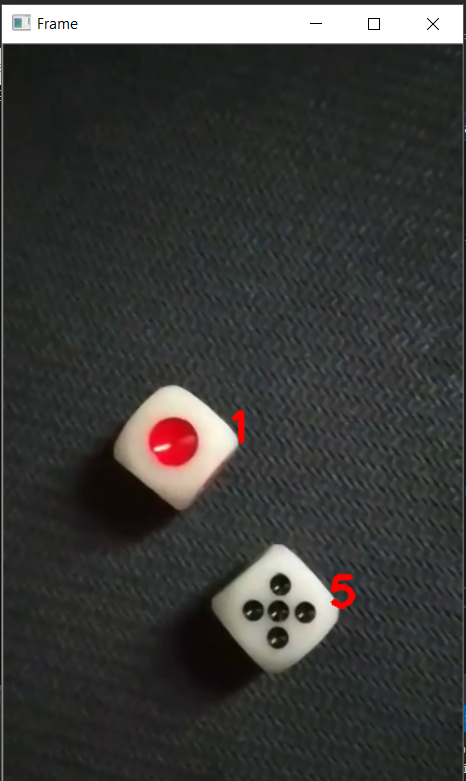
* Video do nhóm tự quay, sử dụng xí ngầu trắng chấm đen, trên nền đen.
* Vì sử dụng điều kiện trên khá thuận lợi cho việc nhận diện, nên nhóm sử dụng cách là đổi frame của video sang nhị phân, sau đó tìm contour.
* Sau khi tìm được contour, ta sử dụng hierarchy để nhận biết cũng như đếm số lượng chấm trên xí ngầu.
* Khi tìm contours, ngoài contours ta còn có một khái niệm khác là hierarchy. Với mỗi contour, ta có một mảng hierarchy tương ứng gồm 4 thành phần tương ứng với chỉ số từ 0 đến 3, cụ thể như sau:



* Nếu một contour A chứa bên trong nó một contour B, thì A là parent của B, B là child của A. Nếu 2 contour không liên quan đến nhau thì gọi là same level.
* Ngoài ra khi tìm contour, ta sắp xếp các contour theo dạng TREE, như vậy các contour sẽ được đánh thứ tự từ ngoài vào trong.



* Dựa vào tính chất trên, ta nhận diện và đếm số chấm như sau:
* Nếu contour đó không có parent, thì đó chính là đường viền bên ngoài của xí ngầu.
* Sau khi nhận diện được contour viền, các contour tiếp theo chính là các chấm của xí ngầu, với mỗi contour, tăng biến đếm lên 1.
* Ta nhận diện chấm cuối cùng bằng phương pháp: nếu contour là chấm cuối cùng, thì contour tiếp theo sẽ là đường viền của xí ngầu khác nghĩa là contour tiếp theo không có parent. Khi nhận diện được chấm cuối cùng, biến đếm chính là tổng số chấm trên mặt xí ngầu.
* **Một số hình ảnh kết quả:**

* Video kết quả nhóm xin được đính kèm trong folder báo cáo, folder **Visual C++**, file **outcpp.avi**

**Chương trình thực hiện**

#include <opencv2/opencv.hpp>

#include <opencv2/highgui.hpp>

#include "opencv2/objdetect.hpp"

#include <iostream>

using namespace cv;

using namespace std;

int main(int argc, char\*\* argv)

{

Mat frame, temp;

Mat gray, binary;

String filename = "video.mp4";

VideoCapture cap(filename);

int frame\_width = cap.get(CV\_CAP\_PROP\_FRAME\_WIDTH);

int frame\_height = cap.get(CV\_CAP\_PROP\_FRAME\_HEIGHT);

VideoWriter video("outcpp.avi", CV\_FOURCC('M', 'J', 'P', 'G'), 30, Size(frame\_width, frame\_height));

if (!cap.isOpened())

throw "Error when reading steam\_avi";

while (1)

{

// Doc tung khung hinh

cap >> frame;

// Neu khong co khung hinh --> stop vong lap

if (frame.empty())

break;

// Loc nhieu, doi sang anh nhi phan

medianBlur(frame, frame, 3);

bilateralFilter(frame, gray, 5, 80, 80);

cvtColor(gray, gray, CV\_BGR2GRAY);

threshold(gray, binary, 100, 255, THRESH\_BINARY);

// Loai bo cac diem sang nho, tach biet cac cham tren xi ngau bang xoi mon,

// gian no

int size = 2;

Mat element = getStructuringElement(MORPH\_RECT , Size(2\*size+1,2\*size+1), Point(size,size));

dilate(binary, binary, element);

erode(binary, binary, element);

size = 1;

element = getStructuringElement(MORPH\_RECT, Size(2 \* size + 1, 2 \* size + 1), Point(size, size));

dilate(binary, binary, element);

// Tim contours

vector <vector<Point> > contours, fil\_contours;

vector <Vec4i> hierarchy;

findContours(binary, contours, hierarchy, CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_SIMPLE, Point(0, 0));

// Quet contour, dem so luong cham den

int dotCount = 0;

int last\_square\_index = 0;

int totalDotCount = 0;

for (int i = 0; i < contours.size(); i++)

{

// Kiem tra contours co phai la lop ngoai cung (vien cua xi ngau)

if (hierarchy[i][3] == -1) {

// Reset bien dem neu tim thay duong vien xi ngau

dotCount = 0;

last\_square\_index = i;

}

else

{

dotCount += 1;

// Kiem tra xem contours co phai la cuoi cung

if ((i == contours.size() - 1) || (hierarchy[i + 1][3] == -1))

{

// Viet so luong cham ben canh xi ngau

stringstream stream;

stream << dotCount;

Moments m = moments(contours[last\_square\_index]);

putText(frame, stream.str(), Point(m.m10 / m.m00 +40, m.m01 / m.m00), FONT\_HERSHEY\_PLAIN, 2.5, Scalar(0,0,255), 3);

}

}

}

// Hien thi ket qua

imshow("Frame", frame);

video.write(frame);

// Press ESC on keyboard to exit

char c = (char)waitKey(25);

if (c == 27)

break;

}

cap.release();

video.release();

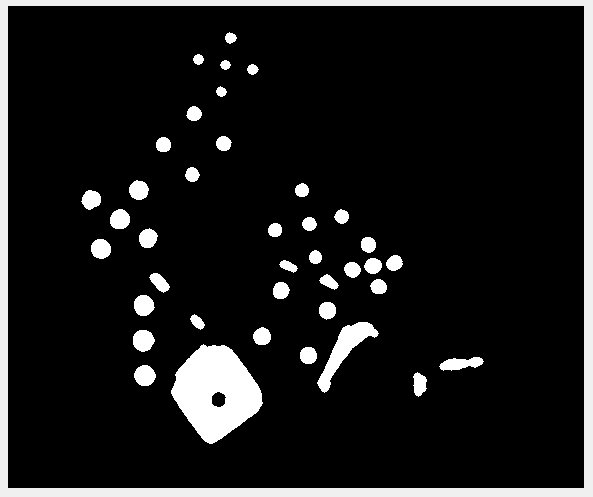
destroyAllWindows();

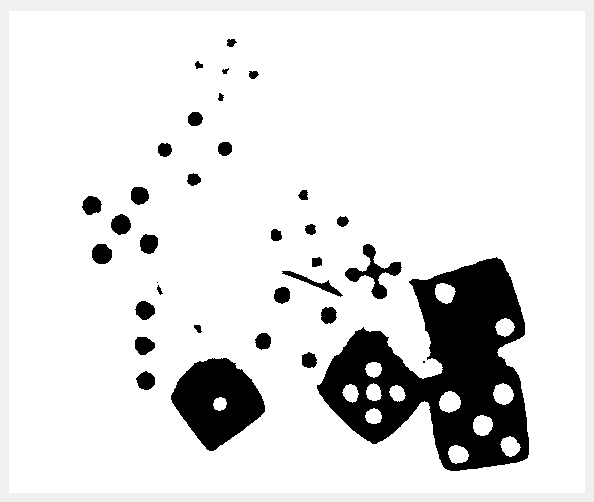
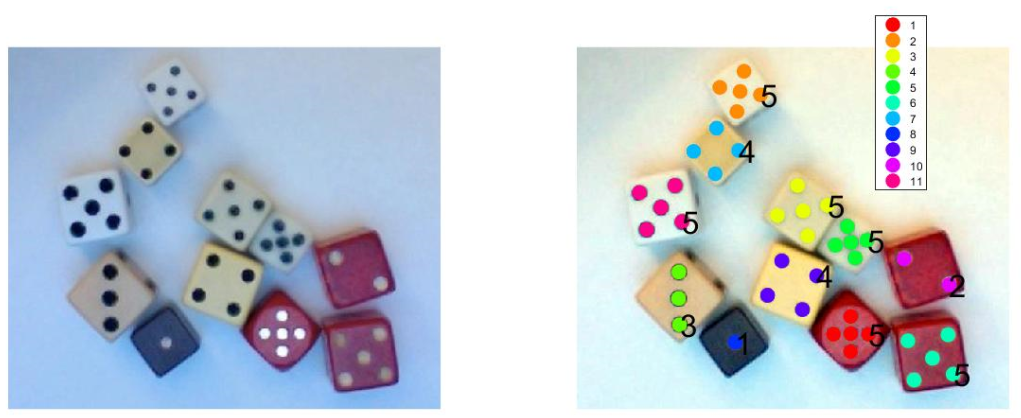
return 0;

}

1. **Chương trình Matlab**
2. **Nhận diện ảnh:**

* Nhóm nhận thấy phương pháp thường sử dụng để nhận diện xí ngầu là đổi ảnh sang nhị phân sau đó tìm contours, dựa vào số contours để đếm số chấm.
* Tuy nhiên, do hình trên tương đối mờ, trong đó có xí ngầu với màu tương đối giống với màu nền (background) nên việc lấy contour hay cạnh một cách chính xác hình dạng của các con xí ngầu là tương đối khó khăn.
* Vì vậy, nhóm sử dụng hàm trong Matlab là regionprops, là hàm tìm những điểm có tính chất giống nhau trong hình, cụ thể là màu sắc. Sử dụng regionprops, ta có thể tìm ra nhóm các điểm có màu đen, hay chấm đen trong hình.
* Vì hình trên gồm 2 loại xí ngầu là chấm đen và chấm trắng, nên nhóm tách làm 2 phần:
* Đầu tiên là đổi ảnh ban đầu sang nhị phân theo ngưỡng V và lấy đảo để tìm xúc xắc trắng chấm đen trước.



* Tiếp đến đổi ảnh ban đầu sang nhị phân theo ngưỡng V và lấy đảo để tìm xúc xắc màu khác.
* Sau khi tìm được các chấm, sử dụng lệnh kmean để phân nhóm các chấm này, các chấm trên cùng một xí ngầu sẽ thành 1 nhóm, cuối cùng ta chỉ cần đếm số chấm trong mỗi nhóm để biết được số chấm trên xí ngầu.
* Tuy nhiên, cách làm này có hạn chế: đó là ta phải biết được số lượng xí ngầu có trong hình, nếu không, việc phân nhóm sẽ bị sai. Đây cũng là hạn chế nhóm chưa khắc phục được.
* Kết quả sau khi thực hiện:
* **Chương trình thực hiện**

clc;

%load the image

I =imread('D:\BT2.png');

soxucxac = 11;

subplot(1,2,1);

imshow(I);

%generate a normalized image

E=I;

E(:,:,1)= imadjust(E(:,:,1));

E(:,:,2)= imadjust(E(:,:,2));

E(:,:,3)= imadjust(E(:,:,3));

%seperate the HSV channels

[H, S, V] = rgb2hsv(E);

B= ~imclose(im2bw(V,.5),strel('disk',4,0));

pips = [];

%find the black pips, them white pips

for temp1 = 1:2

[L,n]= bwlabel(B);

blobs = regionprops(L);

for i = 1:n

c = blobs(i).Centroid;

r = blobs(i).BoundingBox;

if (norm(r(3) - r(4)) <= 2) && (blobs(i).Area > (r(3) + r(4)/2)\*2.5)

radius = (r(3)\*.5 + r(4)\*.5)/2;

if norm(blobs(i).Area - pi\*radius^2) <= ((r(3) +r(4)/2))+ 20

pips(end + 1,:) = c;

end

end

end

B = ~imclose(im2bw(S+H-V,0.35),strel('disk',5,0));

end

[row,col] = size(pips);

idx = kmeans(pips,soxucxac,'Replicates',200);

sum =0;

subplot(1,2,2);

imshow(E)

hold on

gscatter(pips(:,1),pips(:,2),idx,[],[],40)

for temp2 = 1:soxucxac

for temp3 = 1:row

if idx(temp3) == temp2

sum = sum + 1;

x = pips(temp3,1);

y = pips(temp3,2);

end

end

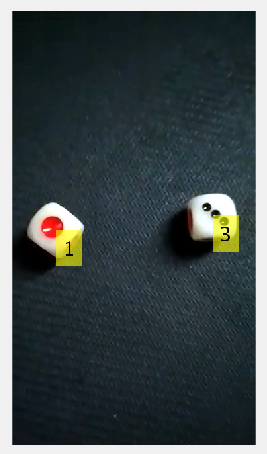
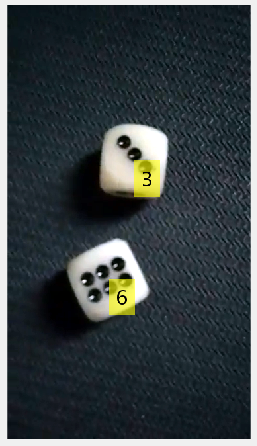
text(x,y,int2str(sum),'color','black','fontsize',25);

sum =0;

end

1. **Nhận diện video**

* Video do nhóm tự quay, sử dụng xí ngầu trắng chấm đen – đỏ, trên nền đen.
* Nhóm tiếp tục sử dụng hàm regionprops để xác định tọa độ, diện tích các khối xúc xắc.
* Tiếp đến sẽ chuyển ảnh sang nhị phân theo Blue để xác định các khối xúc xắc trắng chấm đen trước.
* Sau đó chuyển ảnh sang nhị phân theo H và V để xác định các khối xúc xắc trắng chấm đỏ sau.
* Tương tự như vậy cho mỗi frame hình cho đến khi kết thúc.
* Dựa vào tính chất trên, ta nhận diện và đếm số chấm như sau:
* Dựa vào tọa độ cũng như diện tích của các khối xúc xắc đã tìm được, cũng như tọa độ các chấm đã xác đinh. Ta đếm số lượng chấm trong từng con xúc xắc.
* **Một số hình ảnh kết quả:**



* Video kết quả nhóm xin được đính kèm trong folder báo cáo, folder **Matlab**, file **video1Obj.avi**

**Chương trình thực hiện**

Vid = VideoReader('D:\video1.mp4');

frameratev = Vid.FrameRate;

VidObj = VideoWriter('D:\video1Obj.avi');

open(VidObj);

%read Video from 0.5s

Vid.CurrentTime = 0.5;

while hasFrame(Vid)

I = readFrame(Vid);

%generate a normalized image

E=I;

E(:,:,1)= imadjust(E(:,:,1));

E(:,:,2)= imadjust(E(:,:,2));

E(:,:,3)= imadjust(E(:,:,3));

F = rgb2gray(E);

fig = medfilt2(F);

gray = im2bw(fig,100/255);

S1 = regionprops(gray, 'Area','BoundingBox');

a = [S1.Area];

%find BoundingBox of Dices

idx = find((1000 <= a)&(a <= 18000));

idx = idx';

[He, Se, Ve] = rgb2hsv(E);

pips = [];

B = ~imclose(im2bw(E(:,:,1),0.48),strel('disk',2,0));

for temp1 = 1:2

[L,n]= bwlabel(B);

blobs = regionprops(L);

for i = 1:n

c = blobs(i).Centroid;

r = blobs(i).BoundingBox;

if (norm(r(3) - r(4)) <= 2) && (blobs(i).Area > (r(3) + r(4)/2)\*4)

radius = (r(3)\*.5 + r(4)\*.5)/2;

if norm(blobs(i).Area - pi\*radius^2) <= ((r(3) +r(4)/2))+ 40

pips(end + 1,:) = c;

end

end

end

B = imclose(im2bw(Se/2+Ve/2,0.65),strel('disk',1,0));

end

for w = 1:size(idx)

sum = 0;

rxx = S1(idx(w)).BoundingBox;

for j = 1:size(pips)

if (pips(j,1) >= rxx(1))&& (pips(j,1) <= (rxx(1)+ rxx(3)))&&...

(pips(j,2) >= rxx(2))&& (pips(j,2)<= (rxx(2)+rxx(4)))

sum = sum +1;

end

end

E = insertText(E,[rxx(1)+rxx(3)/2 rxx(2)+rxx(4)/2],int2str(sum),'FontSize',30);

end

%Exe frame 5th in 5 frames

pause(5/frameratev);

writeVideo(VidObj,E);

end

close(VidObj);