**Day 4**

Image transition

Hough transform

Shape detection in a video

**CODES**

Rotated Video

#include "stdafx.h"

#include "opencv2/core/core.hpp"

#include "opencv2/highgui/highgui.hpp"

#include <stdio.h>

#include "opencv2/imgproc/imgproc.hpp"

#include <iostream>

#include <math.h>

using namespace cv;

int main(int argc, \_TCHAR\* argv[])

{

Mat frame;

namedWindow("Say Hi", 1);

VideoCapture cap(0);

cap >> frame;

Mat out;

int slidervalue = 0;

Point pt;

pt.x = frame.rows / 2;

pt.y = frame.cols / 2;

createTrackbar("Rotation", "Say Hi", &slidervalue, 360);

while (1){

cap >> frame;

Mat matRotation = getRotationMatrix2D(pt, slidervalue, 1);

Mat imgRotated;

warpAffine(frame, imgRotated, matRotation, frame.size());

out = imgRotated.clone();

imshow("Say Hi", out);

if (waitKey(30) == 27) break;

}

return 0;

}

Video Fun

#include "stdafx.h"

#include "opencv2/core/core.hpp"

#include "opencv2/highgui/highgui.hpp"

#include <stdio.h>

#include "opencv2/imgproc/imgproc.hpp"

#include <iostream>

#include <math.h>

using namespace cv;

int main(int argc, \_TCHAR\* argv[])

{

Mat frame;

namedWindow("Say Hi", 1);

VideoCapture cap(0);

Mat out;

int slidervalue = 0;

Point pt;

createTrackbar("Rotation", "Say Hi", &slidervalue, 360);

int x = 0, i, j,y=0;

int temp;

while (1){

cap >> frame;

out = frame.clone();

for (x = 0; x < 3; x++){

for (i = 0; i < frame.rows; i++){

for (j = 0; j < frame.cols / 2; j++) {

temp = out.at<Vec3b>(i, j)[x];

out.at<Vec3b>(i, j)[x] = out.at<Vec3b>(i, frame.cols - j)[x];

out.at<Vec3b>(i, frame.cols - j)[x] = temp;

}

}

}

pt.x = out.cols/2;

pt.y = out.rows/2;

Mat matRotation = getRotationMatrix2D(pt, slidervalue, 1);

Mat imgRotated;

warpAffine(out, imgRotated, matRotation, frame.size());

out = imgRotated.clone();

for (i = 0; i < out.rows; i++){

for (j = 0; j < out.cols ; j++) {

out.at<Vec3b>(i, j)[y]=200;

}

}

y++;

if (y == 3) y = 0;

Mat out1;

Canny(out, out1, 100, 150, 3, false);

imshow("Say Hi", out1);

if (waitKey(30) == 27) break;

}

return 0;

}

Transition of video to image

#include "stdafx.h"

#include "opencv2/core/core.hpp"

#include "opencv2/highgui/highgui.hpp"

#include <stdio.h>

#include "opencv2/imgproc/imgproc.hpp"

#include <iostream>

#include <math.h>

using namespace cv;

int main(int argc, \_TCHAR\* argv[])

{

namedWindow("See the Change to Be change", 1);

int value = 0;

int i, j, x;

VideoCapture cap(0);

createTrackbar("Changer", "See the Change to Be change", &value, 100);

while (1){

Mat img3;

cap >> img3;

Mat img1;

Mat img2 = imread("def.jpg", 1);

resize(img3, img1, img2.size());

Mat img = img2.clone();

for (x = 0; x < 3; x++){

for (i = 0; i < img.rows; i++){

for (j = 0; j < img.cols; j++){

img.at<Vec3b>(i, j)[x] = img2.at<Vec3b>(i, j)[x] \* (value / 100.0) + img1.at<Vec3b>(i, j)[x] \* ((100-value) / 100.0);

}

}

}

imshow("See the Change to Be change", img);

if (waitKey(30) == 27) break;

}

return 0;

}

Longest Path

#include "stdafx.h"

#include "opencv2/core/core.hpp"

#include "opencv2/highgui/highgui.hpp"

#include <stdio.h>

#include "opencv2/imgproc/imgproc.hpp"

#include <iostream>

#include <math.h>

using namespace cv;

int main(int argc, \_TCHAR\* argv[])

{

VideoCapture cap(0);

namedWindow("canny", 1);

int value = 100;

createTrackbar("Threshold", "canny", &value, 255);

int i, j, k,i1,j1;

while (1){

Mat img;

cap >> img;

Mat img1;

Canny(img, img1, 120, 200, 3, false);

Mat plot(10000,400,CV\_8UC1,Scalar(0));

for (i = 0; i < img1.rows; i++){

for (j = 0; j < img1.cols; j++){

if (img1.at<uchar>(i, j)==255) {

for (k = 0; k < 360; k++) plot.at<uchar>((1000+(int)(j\*cos(k\*3.14 / 180.0) + i\*sin(k\*3.14 / 180.0)) ), k)++;

}

}

}

k = 0;

Mat img2(img1.rows, img1.cols, CV\_8UC1);

img2 = Scalar(0);

for (i = 0; i < 1400; i++){

for (j = 0; j < 180; j++){

if (plot.at<uchar>(i, j) > k && plot.at<uchar>(i,j)<value) {

k = plot.at<uchar>(i, j);

i1 = i;

j1 = j;

}

}

}

for (i = 0; i < img2.rows; i++){

for (j = 0; j < img2.cols; j++){

if (i1 == (1000 + (int)(j\*cos(j1\*3.14 / 180.0) + i\*sin(j1\*3.14 / 180.0)))) {

if (img1.at<uchar>(i, j) == 255){

img2.at<uchar>(i, j) == 255;

circle(img2, Point(j, i), 1, 255, 2, 8, 0);

}

}

}

}

imshow("Longest path", img2);

imshow("canny", plot);

if (waitKey(50) == 27) break;

}

return 0;

}

Hough circles

#include "stdafx.h"

#include "opencv2/core/core.hpp"

#include "opencv2/highgui/highgui.hpp"

#include <stdio.h>

#include "opencv2/imgproc/imgproc.hpp"

#include <iostream>

#include <math.h>

using namespace cv;

int main(int argc, \_TCHAR\* argv[])

{

VideoCapture cap(0);

while (1){

vector<Vec3f> circles;

Mat img;

Mat img1;

cap >> img1;

cvtColor(img1, img1, CV\_RGB2GRAY);

GaussianBlur(img1, img, Size(9, 9), 2, 2);

HoughCircles(img, circles, CV\_HOUGH\_GRADIENT, 1, img.rows / 16, 200, 30, 0, 0);

for (size\_t i = 0; i < circles.size(); i++)

{

Point center(cvRound(circles[i][0]), cvRound(circles[i][1]));

int radius = cvRound(circles[i][2]);

circle(img, center, 3, Scalar(0, 255, 0), -1, 8, 0);

circle(img, center, radius, Scalar(0, 0, 255), 3, 8, 0);

}

namedWindow("Hough Circle Transform Demo", CV\_WINDOW\_AUTOSIZE);

imshow("Hough Circle Transform Demo", img);

if (waitKey(30) == 27) break;

}

return 0;

}

Video Shape Detection

#include "stdafx.h"

#include "opencv2/core/core.hpp"

#include "opencv2/highgui/highgui.hpp"

#include <stdio.h>

#include "opencv2/imgproc/imgproc.hpp"

#include <iostream>

#include <math.h>

using namespace cv;

int main(int argc, \_TCHAR\* argv[])

{

int i,a=10,j,b=10,c=10,a1=100,b1=100,c1=100;

VideoCapture cap(0);

namedWindow("Contour", 1);

createTrackbar("Color\_blue1", "Contour", &a, 255);

createTrackbar("Color\_green1", "Contour", &b, 255);

createTrackbar("Color\_red1", "Contour", &c, 255);

createTrackbar("Color\_blue2", "Contour", &a1, 255);

createTrackbar("Color\_green2", "Contour", &b1, 255);

createTrackbar("Color\_red2", "Contour", &c1, 255);

while (1){

Mat img;

cap >> img;

Mat img2 = img.clone();

Mat img1;

cvtColor(img, img1, CV\_BGR2HLS);

vector<Mat> channel;

split(img1, channel);

for (i = 0; i < img.rows; i++){

for (j = 0; j < img.cols; j++){

if (channel[0].at<uchar>(i, j)<a1 && channel[0].at<uchar>(i, j)>a) {

if (channel[1].at<uchar>(i, j)<b1 && channel[1].at<uchar>(i, j)>b) {

if (channel[2].at<uchar>(i, j)<c1 && channel[2].at<uchar>(i, j)>c) {

img.at<Vec3b>(i, j) = { 255, 255, 255 };

}

else img.at<Vec3b>(i, j) = { 0, 0, 0 };

}

else img.at<Vec3b>(i, j) = { 0, 0, 0 };

}

else img.at<Vec3b>(i, j) = { 0, 0, 0 };

}

}

/\*Canny(img, img, 120, 200, 3);\*/

imshow("Contour", img);

if (waitKey(30) == 27) {

i = 3;

while (i--){

Canny(img, img, 150, 200, 3);

blur(img, img, Size(3, 3));

vector<vector<Point>> contours;

vector<Vec4i> hierarchy;

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

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

drawContours(img2, contours, i, Scalar(0, 0, 0), 2, 8, hierarchy, 0, Point());

}

imshow("Contour", img2);

waitKey(0);

}

}

}

return 0;

}