1：

opencv+arduino进行物体点追踪

wi162yyxq 2016-08-30 15:10:31 10242 收藏 26

分类专栏： opncv

版权

本文所要实现的结果是：通过在摄像头中选择一个追踪点，通过pc控制摄像头的舵机，使这一点始终在图像的中心。

要点：使用光流法在舵机旋转的同时进行追踪，若该点运动，则摄像头跟踪联动。

#include<opencv2\opencv.hpp>

#include<opencv\cv.h>

#include<opencv\highgui.h>

#include<math.h>

#include<Windows.h>

#include<string.h>

using namespace std;

using namespace cv;

#define WINDOW\_NAME "【程序窗口】"

void on\_MouseHandle(int event, int x, int y, int flags, void\* param);

void DrawRectangle( cv::Mat& img, cv::Rect box );

void tracking(Mat &frame,vector<Point2f> temp);

HANDLE hComm;

LPCWSTR pStr=L"COM4";

char lpOutbuffer[100];

DWORD dwbyte=100;

Mat srcImage,grayImage,tempImage1,tempImage,imageROI,grayprev;

int g\_maxCornerNumber = 1;

double qualityLevel = 0.01;

double minDistance = 10;

int blockSize = 3;

double k = 0.04;

vector<Point2f> corners;

vector<Point2f> pre\_corners;

vector<Point2f> counts;

vector<uchar> status;

vector<float> err;

Rect g\_rectangle;

Rect g\_temprectangle;

bool g\_bDrawingBox = false;

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

{

Mat frame;

Mat result;

COMSTAT Comstat;

DWORD dwError;

BOOL bWritestat;

hComm=CreateFile(pStr,GENERIC\_READ | GENERIC\_WRITE,0,0,OPEN\_EXISTING, 0,NULL);

if (hComm == INVALID\_HANDLE\_VALUE)

{

cout<<"FLASE";

return -1;

}

else

{

cout<<"TURE";

}

DCB dcb;

GetCommState(hComm,&dcb);

dcb.BaudRate=9600;

dcb.ByteSize=8;

dcb.Parity=NOPARITY;

dcb.StopBits=TWOSTOPBITS;

bool set=SetCommState(hComm,&dcb);

bool sup=SetupComm(hComm,1024,1024);

VideoCapture capture(0);

namedWindow( WINDOW\_NAME );

setMouseCallback(WINDOW\_NAME,on\_MouseHandle,(void\*)&frame);

while(1)

{

capture >> frame;

if(!frame.empty())

{

cvtColor(frame,grayImage,CV\_RGB2GRAY);

if( g\_bDrawingBox )

rectangle(frame,g\_rectangle.tl(),g\_rectangle.br(),Scalar(255,255,255));

if (corners.size()!=0)

{

bool can=PurgeComm(hComm,PURGE\_TXCLEAR);

if (corners[0].x>(frame.cols/2+100))

{

lpOutbuffer[0]='a';

bool ne=WriteFile(hComm,lpOutbuffer,dwbyte,&dwbyte,NULL);

}

else if (corners[0].x<(frame.cols/2-100))

{

lpOutbuffer[0]='b';

bool ne=WriteFile(hComm,lpOutbuffer,dwbyte,&dwbyte,NULL);

}

tracking(frame,corners);

rectangle(frame,Point(corners[0].x-10,corners[0].y-10),Point(corners[0].x+10,corners[0].y+10),Scalar(255,255,255));

}

imshow( WINDOW\_NAME, frame );

}

else

{

printf(" --(!) No captured frame -- Break!");

break;

}

int c = waitKey(50);

if( (char)c == 27 )

{

break;

}

}

return 0;

}

void on\_MouseHandle(int event, int x, int y, int flags, void\* param)

{

Mat& image = \*(cv::Mat\*) param;

switch( event)

{

case EVENT\_MOUSEMOVE:

{

if( g\_bDrawingBox )

{

g\_rectangle.width = x-g\_rectangle.x;

g\_rectangle.height = y-g\_rectangle.y;

}

}

break;

case EVENT\_LBUTTONDOWN:

{

g\_bDrawingBox = true;

g\_rectangle =Rect( x, y, 0, 0 );

}

break;

case EVENT\_LBUTTONUP:

{

g\_bDrawingBox = false;

if( g\_rectangle.width < 0 )

{

g\_rectangle.x += g\_rectangle.width;

g\_rectangle.width \*= -1;

}

if( g\_rectangle.height < 0 )

{

g\_rectangle.y += g\_rectangle.height;

g\_rectangle.height \*= -1;

}

imageROI=grayImage(g\_rectangle);

goodFeaturesToTrack( imageROI,corners,g\_maxCornerNumber,qualityLevel,minDistance,Mat(),blockSize,false,k );

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

{

corners[i].x=corners[i].x+g\_rectangle.x;

corners[i].y=corners[i].y+g\_rectangle.y;

}

}

break;

}

}

void tracking(Mat &frame,vector<Point2f> temp)

{

cvtColor(frame, tempImage1, COLOR\_BGR2GRAY);

if (grayprev.empty())

{

tempImage1.copyTo(grayprev);

}

calcOpticalFlowPyrLK(grayprev, tempImage1, temp, pre\_corners, status, err);

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

{

line(frame, temp[i], pre\_corners[i], Scalar(0, 0, 255));

circle(frame, pre\_corners[i], 4, Scalar(0, 255, 0), -1,8,0);

}

swap(pre\_corners, corners);

swap(grayprev, tempImage1);

}