EPH - International Journal Of Science And Engineering
ISSN (Online): 2454 2016
Volume 07 Issue 03 September 2021

DOI:https://doi.org/10.53555/ephijse.v7i3.187

RESEARCH ON TARGET RECOGNITION OF COMBAT ROBOT BASED ON OPENCV

Jin Xu[1*]
[*1]*School of Electrical and Information, Jiangsu University of Science and Technology, Zhenjiang, China, 212003*

Corresponding Author:-

Abstract:-
With the wide application of robots in various fields of human life, the research on improving the level of intelligent robot is highly concerned by domestic and foreign scholars. As an important part of intelligent robot, robot vision has been paid more and more attention in recent years. Computer vision technology is the use of image processing technology to make the camera and the human eye to identify, judge, feature detection, tracking and other visual functions. The computer vision system is to create a complete artificial intelligence system which can get the needed information in the plane image or the three-dimensional image data. Opencv plays an important role in the field of computer vision which is a cross platform computer vision library based on open source. The main purpose of this study is: According to the feature recognition and matching of the object, make the camera of the combat robot has the vision through a detection system based on OpenCV visual library to detect and track the moving objects in the line of sight

Keywords:- *computer vision; feature detection; image processing; Opencv*

I INTRODUCTION

Computer vision is a technology field which has been improved in recent years. With the development of electronic circuits and the advent of high-performance computer devices, it becomes more and easier to create complex image applications. Computer vision is a machine learning how to "see", specifically, is to identify the target by using cameras and computer instead of human eyes, tracking reaction and machine vision, and further image processing, making it more suitable for machine detection technology.

The full name of Opencv is Open Source Computer Vision Library. Opencv , a cross platform based on open source computer vision library which you can run on Linux, Windows, Mac, Android, IOS, Os, Maemo, FreeBSD, OpenBSD and other operating systems.,was founded in 1999 by Intel and is now supported by Willow Garage. Opencv consists of a series of C functions and C++ class, with more than and 500 C functions, including cross platform in the high-level API. Opencv has become one of the most powerful research tools in the field of computer vision. It can make full use of the advantages of high performance multi-core processor to build a simple and easy computer vision framework.

In this paper, I establish the target recognition model of combat robot based on Opencv image processing technology. In the Visual Studio2010 development environment, read the object through the camera, and binarization separation characteristics of color of the object, using the image processing function in Opencv ((findContours) (contour search), draw the outline (drawContours) (), return external rectangular boundary (boundingRect) (), with two-dimensional ellipse fitting set fitEllipse ()) to fit the image contour optimization. If that meet the requirements of the feature, the target was found, then the motherboard send instructions to the pan and launch bullets.

II The basic principle of picture binarization

The binarization of the image is the gray value of the point on the image is 0 or 255, that is, the whole image showing a significant black and white effect. A 256-brightness grayscale image is selected by appropriate threshold selection to obtain a binarized image that still reflects the overall and local features of the image. In order to obtain the ideal binary image, we generally use closed, connected boundary definition of non-overlapping areas. All pixels whose gray levels are greater than or equal to the thresholds are determined to belong to a particular object with a gray value of 255, otherwise the pixels are excluded from the object region and have a gray value of 0, representing the background or exception of the object area. This image segmentation method is based on the gray difference between the object and the background in the image, and this segmentation belongs to the pixel level segmentation. If a particular object has a uniform gray value inside it, and it is in a uniform background with other grayscale values, the comparison effect can be obtained using the threshold method. If the difference between an object and the background is not on a grayscale value (such as a different texture), the difference feature can be converted to a gray scale difference, and then the image is segmented using the threshold selection technique. Dynamically adjust the threshold to achieve the binarization of the image can be dynamically observed its specific results of the split image.

III Use Opencv to achieve picture binarization

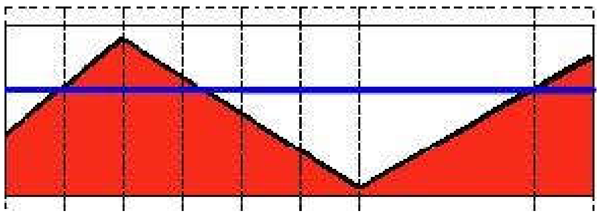
In Opencv, the Threshold () function (basic threshold operation) and the adaptive Threshold () function (adaptive threshold operation) can complete the binarization of the picture. Their basic idea is to give an array and a threshold, and then do some processing according to the array of each element value is higher or lower than the threshold.

The function Threshold () applies a fixed threshold operation to a single-channel array. The typical application of this function is to perform a threshold operation on the grayscale image to get a binary image or remove the noise. The function prototype is as follows:

C++：double threshold (InpuArray src,OutoutArray dst,double thresh,double maxval,int type)

1. The first parameter，InputArray type src, enter the array, fill the single channel, 8 or 32-bit floating point type of Mat can be.
2. The second parameter，The outputstray type dst stores the result of the function and has the same size and type as the Mat variable in the first argument。
3. The third parameter，Double type thresh, the specific value of the threshold.
4. The fourth parameter，Double type maxval, when the fifth parameter threshold type type takes the maximum value when the threshold type is CV-THRESH-BINARY or CV-THRESH-BINARY-INV.
5. The fifth parameter，Int type of type, type of threshold, The Threshold () function supports a method of taking a threshold for an image.

In order to explain the process of threshold segmentation, let's look at a simple picture of the gray scale of the pixel. The blue horizontal line in the figure represents a specific threshold.

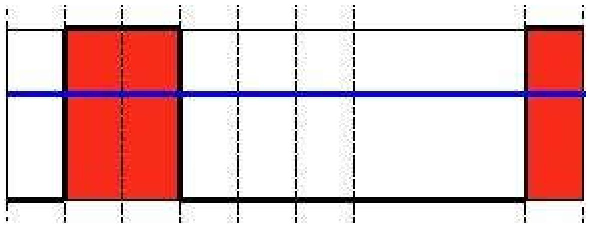


Threshold Type 1: Binary Thresholding

- The threshold type is as follows:
-

$$\mathtt{dst}(x,y) = \begin{cases} \mathtt{maxVal} & \text{if } \mathtt{src}(x,y) > \mathtt{thresh} \\ 0 & \text{otherwise} \end{cases}$$

- Explanation: When using this threshold type, a specific threshold amount is selected, such as 125, so that the new threshold generation rule can be interpreted as a gray value of a pixel that is greater than 125 is set to a maximum and a gray value of a pixel with a gray scale value of less than 125 is set to zero。
-

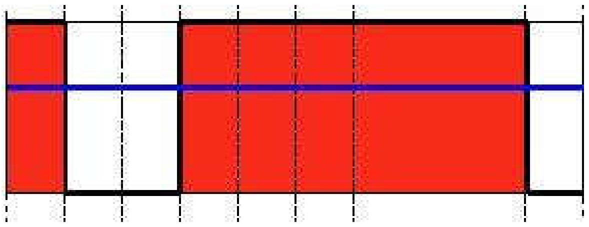


Threshold Type 2: Anti-Binary Thresholding

- The threshold type is as follows:

$$\mathtt{dst}(x,y) = \begin{cases} 0 & \text{if } \mathtt{src}(x,y) > \mathtt{thresh} \\ \mathtt{maxVal} & \text{otherwise} \end{cases}$$

-
- Explanation: This threshold is similar to binary threshold, first selecting a specific gray value as the threshold, but the last setpoint is reversed.

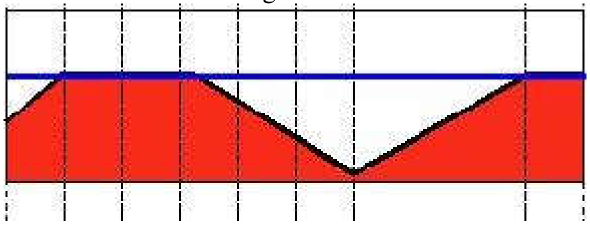


Threshold Type 3: Truncate Threshold

- The threshold type is as follows:
-

$$\mathtt{dst}(x,y) = \begin{cases} \mathtt{threshold} & \text{if } \mathtt{src}(x,y) > \mathtt{thresh} \\ \mathtt{src}(x,y) & \text{otherwise} \end{cases}$$

- Explanation: It is also necessary to first select a threshold in which the pixel above the threshold is set to the threshold, but that less than the threshold remains unchanged.

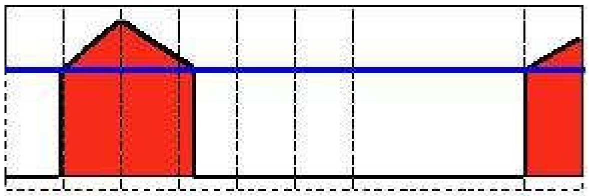


Threshold Type 4: Thresholds to 0

- The threshold type is as follows:

$$\mathtt{dst}(x,y) = \begin{cases} \mathtt{src}(x,y) & \text{if } \mathtt{src}(x,y) > \mathtt{thresh} \\ 0 & \text{otherwise} \end{cases}$$

- Explanation: A threshold is selected first, and then the image is processed as follows: The gray value of the pixel greater than the threshold remains unchanged; If the gray value of the pixel is less than the threshold, the gray value all become zero.

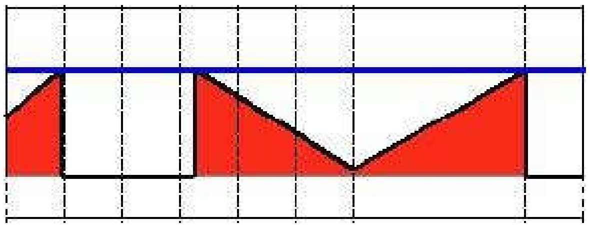


Threshold Type 5: Anti-Threshold is 0

- The threshold type is as follows:

$$\mathrm{dst}(x,y) = \begin{cases} 0 & \text{if } \mathrm{src}(x,y) > \mathrm{thresh} \\ \mathrm{src}(x,y) & \text{otherwise} \end{cases}$$

- Explanation: The principle is similar to the 0 threshold, but the opposite is done when the image is processed. That is, The gray value of the pixel less than the threshold remains unchanged; If the gray value of the pixel is greater than the threshold, the gray value all become zero.
-

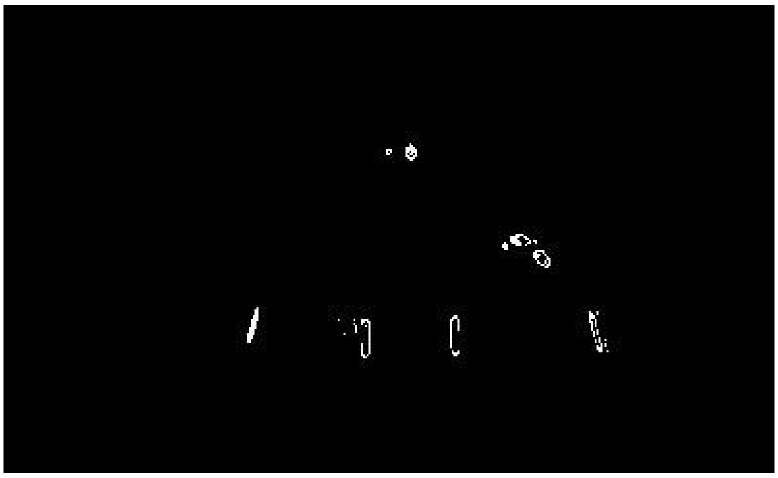


The adaptiveThreshold () function takes an adaptive threshold operation on the matrix and supports in-place operations. The function prototype is as follows:

C++: void adaptiveThreshold(InputArray src,OutputArray dst,double maxValue,int adaptiveMethod,int thresholdType,int blockSize,double C)

1. The first parameter，InputArray type src, enter the image, that is, the source image, filling the Mat class object, and need 8-bit single-channel floating-point images.
2. The second parameter, OutputArray type dst, which stored the results of the function ,need to have the same size and type as the source image.
3. The third parameter，Double of the type of maxValue, given the pixel to meet the conditions of the non-zero value.
4. The fourth parameter，The adaptiveMethod for the int type specifies the adaptive threshold algorithm to use.
5. The fifth parameter，Int type thresholdType, threshold type. Value must come from THRESH_BINARY or THRESH_BINARY_INV.
6. The sixth parameter，Int type of blockSize, used to calculate the threshold size of a pixel size of the neighborhood.
7. The seventh parameter，Double type C, the constant value which minus the average or weighted average. Usually it is positive, and in a few cases it is zero or negative.

This article uses the Threshold () function, the effect is as follows：



The original picture：

IV Extract and plot the area image outline

The pictures read by the camera usually contain various objects and one of the purposes of image analysis is to identify and extract these objects. In the object detection and recognition program, the first step is to generate a binary image to get the location of the target object and the next step is from 1 and 0 composed of pixels from the collection of objects. Opencv can be used to find the outline from the binary image using the findContours () function, The function prototype is as follows：

C++ ： void findContours(InputOutputArray image,OutputArrayOfArrays contours,OutputArray hierarchy,int mode,int method,Point offset=Point())

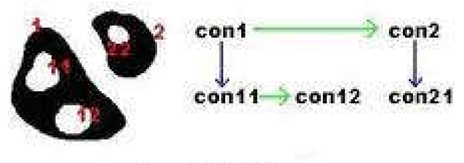
1. The first parameter，InputArray type image, input image that is source image. Non-zero pixels of the image are treated as 1,0 pixels are reserved for 0, that is the image is binary.
2. The second parameter， OutputArrayOfArrays type contours, detected contours, save the result of the operation after the function call. Each contour is stored as a point vector, that is, a vector of type point.
3. The third parameter，OutputArray type of hierarchy, optional output vector, including the topology of the image information. As a representation of the number of contours, contains a lot of elements. Each contour contours [i] correspond to the four hierarchy elements [i] [0] to hierarchy [i] [3], which represent the index number of the next contour, the previous contour, the parent contour, and the embedded contour, respectively. If there is no corresponding item, the corresponding hierarchy [i] value is set to a negative number.
4. The fourth parameter，Int type of mode, contour search mode.The values are as follows:

RETR_EXTERNAL： indicates that only the outermost contours are detected.

RETE_LIST：Extract all the contours and place them in the list.

RETR_CCOMP：Extract all contours and organize them into a two-layer structure. The first layer is the outer boundary of all objects. The second layer is the boundary of all holes. If there are other connected objects in the hole, the object is divided into the first layer.

RETR_TREE：Extract all the contours, and re-establish the mesh contour structure.



OpenCV 轮廓检测
CV_RETR_TREE参数示意图

5. The fifth parameter，Int type of method, for the outline of the approximate approach, the values are as follows:

CHAIN_APPROX_NONE：Gets each pixel of each contour, the adjacent two points of the pixel position difference does not exceed 1.

CHAIN_APPROX_SIMPLE ： compresses the horizontal, vertical, and diagonal elements, leaving only the end coordinates of the direction.

CHAIN_APPROX_TC89_L1，CHAIN_APPROX_TC89_KCOS：Using one of the Teh-Chinl chain approximation algorithms.

Call example: Vector<vector<Point>> contours;
findContours(image,

Contours, // Outline array
CV_RETR_EXTERNAL, // Get outside contours
CV_CHAIN_APPROX_NONE) ;// Gets each pixel for each contour

Using the findContours () function to detect the outline of the image, drawing the outline will need another function - drawContours () to match. The following describes the drawContours () function.
The drawContours () function is used to draw an external or internal contour in an image whose prototype is as follows:
C++： void drawContours(InputOutputArray image,InputArrayOfArrays contours,int contourIdx,const Scalar&color,int thickness=1,int lineType=8,InputArray hierarchy=noArray(),int maxLevel=INT_MAX,Point offset=Point())

1. The first argument，InputArray type image, said input image, the function will be drawn on this image contour.
2. The second parameter，InputArrayOfArrays type contours that represent pointers to the outline list.
3. The third parameter，Int type contourIdx, the outline of the specified variable.
4. The fourth parameter，Const Scalar & type of color, the color of the outline.
5. The fifth parameter，Indicates the width of the contour, and if it is CV_FILLED it will fill the inside of the contour.
6. The sixth parameter，Int type lineType, type of line.
7. The seventh parameter，InputArray type of hierarchy, optional hierarchical information.
8. The eighth parameter，Int type maxLevel, that the maximum number of layers to draw the outline, if it is 0, only draw contour; If it is 1, additional drawing and contour of all the contours of the same layer; If the value is negative, the function does not draw the contour after the contour, but it will draw its sub-profile until abs (max_level) - 1 (if the value is negative) Floor.
9. The ninth parameter，The offset of type OF indicates the offset, offset by the specified offset offset = (dx, dy), with the default value of Point ().

Call example:
// Draw a black outline
Mat result (image.size(),CV_8U,cv::Scalar(255)); drawContours(result,contours,
-1, // Draw all contours
Scalar (0), // Draw the color in black
3) ； // the line width of the contour line is 3

V Use a polygon to surround the contour

As the combat robot in this paper is mainly to identify the lamp as the characteristics of the target, it is also necessary to display the outline of the image read by the camera with polygons.
Here are some of the functions that are commonly used when creating a polygon boundary that surrounds an outline with Opencv：

- boundingRect()：

This function calculates and returns the rectangle boundary of the specified point set up-right. The function prototype is ：
C++：Rect boundingRect(InputArray points)
Its only one parameter is the input of the two-dimensional point set.

- minEnclosingCircle()：

The function of the minEnclosingCircle function is to use a iterative algorithm to find the smallest circular area that can surround them for a given set of 2D points. The function prototype is:
C++：void minEnclosingCircle(InputArray points,Point2f& center,float& radius)
1．The first parameter ， InputArray type of points, the input of the two-dimensional point set, can be std :: vector <> or Mat type.
2．The second parameter，Point2f & type of center, circle output center.
3．The third parameter，Float & type radius, circle output radius.

- fitEllipse()：

This function is used to fit a two-dimensional point set with ellipse.
C++：RotatedRect fitEllipse(InputArray points)
Its only one parameter is the input of the two-dimensional point set.

- approxPolyDP():

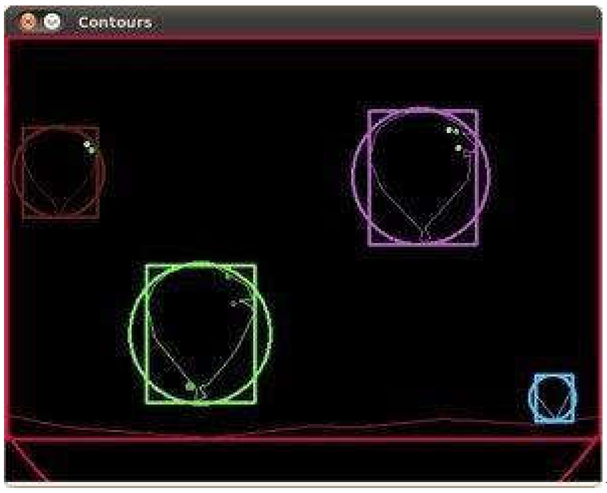
The function of this function is to approximate the polygon curve with the specified precision.
C++：void approxPolyDP(InputArray curve,OutputArray approxCurve,double epsilon,bool closed)
1．The first parameter，InputArray type of curve, enter the two-dimensional point set.
2．The second parameter，OutputArray type approxCurve, the result of the polygon approximation, whose type should be consistent with the type of the input two-dimensional dot set.
3．The third parameter，Double type epsilon, which specifies the parameters of the estimated precision.

4．The fourth parameter，Bool type of closed, if true: the estimated curve is closed, if it is false: the estimated curve is not closed.
This article uses the Opencv function boundingRect () to compute the rectangular box of the enclosing contour, using minEnclosingCircle () to compute the absolute circle that surrounds the existing contour. The code is described below：

```
#include"opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>

usingnamespacecv;
usingnamespacestd;

```

```
Mat src; Mat src_gray;
int thresh = 100;
int max_thresh = 255;
RNG rng(12345);

/// 函数声明
void thresh_callback(int, void* );

/** @主函数 */
int main( int argc, char** argv )
{
  /// 载入原图像, 返回 3 通道图像
  src = imread( argv[1], 1 );

  /// 转化成灰度图像并进行平滑
  cvtColor( src, src_gray, CV_BGR2GRAY );
  blur( src_gray, src_gray, Size(3,3) );

  /// 创建窗口
  char* source_window = "Source";
  namedWindow( source_window, CV_WINDOW_AUTOSIZE );
  imshow( source_window, src );

  createTrackbar( " Threshold:", "Source", &thresh, max_thresh, thresh_callback );
  thresh_callback( 0, 0 );

  waitKey(0);
  return(0);
}

/** @thresh_callback 函数 */
void thresh_callback(int, void* )
{
  Mat threshold_output;
  vector<vector<Point> > contours;
  vector<Vec4i> hierarchy;

  /// 使用 Threshold 检测边缘
  threshold( src_gray, threshold_output, thresh, 255, THRESH_BINARY );
  /// 找到轮廓
  findContours( threshold_output, contours, hierarchy, CV_RETR_TREE, CV_CHAIN_APPROX_S
IMPLE, Point(0, 0) );

  /// 多边形逼近轮廓 + 获取矩形和圆形边界框
  vector<vector<Point> > contours_poly( contours.size() );
  vector<Rect> boundRect( contours.size() );
  vector<Point2f>center( contours.size() );
  vector< float >radius( contours.size() );

  for( int i = 0; i < contours.size(); i++ )
     { approxPolyDP( Mat(contours[i]), contours_poly[i], 3,        true );
       boundRect[i] = boundingRect( Mat(contours_poly[i]) );
       minEnclosingCircle( contours_poly[i], center[i], radius[i] );
     }


  /// 画多边形轮廓 + 包围的矩形框 + 圆形框
  Mat drawing = Mat::zeros( threshold_output.size(), CV_8UC3 );
  for( int i = 0; i< contours.size(); i++ )
     {
       Scalar color = Scalar( rng.uniform(0, 255), rng.uniform(0,255), rng.uniform(0,2
55) ) ;
       drawContours( drawing, contours_poly, i, color, 1, 8, vector<Vec4i>(), 0, Point
() );
       rectangle( drawing, boundRect[i].tl(), boundRect[i].br(), color, 2, 8, 0 );
       circle( drawing, center[i], (     int )radius[i], color, 2, 8, 0 );
     }

  /// 显示在一个窗口
  namedWindow(   "Contours"  , CV_WINDOW_AUTOSIZE );
  imshow(   "Contours"  , drawing );
}</span>
```

The operating results are as follows:



VI Finished product display and analysis

This article builds the combat robot's physical structure as follows, the white ball figure 1 is bullet. Figure 2 shows the red rays of the pointing area to the target to be hit. Figure 3 simulates the battle robot to find the target, and then implement the strike process. Experiments show that the host computer through the camera to identify the target, the next crew to send instructions correctly, you can implement the target strike.

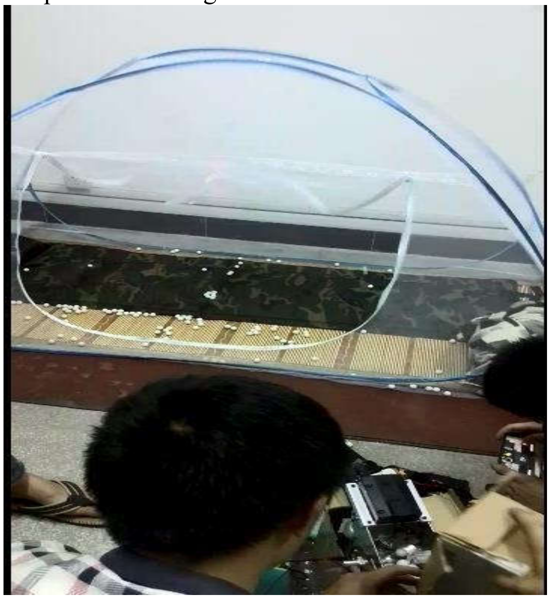


Figure 1 physical map of the launch mechanism

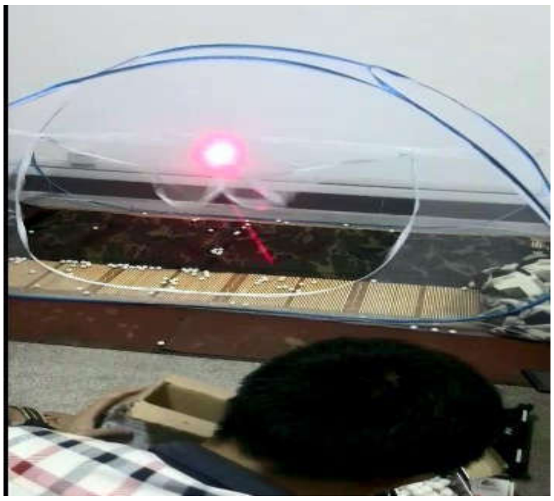


Figure 2 launch mechanism to launch a bullet chart

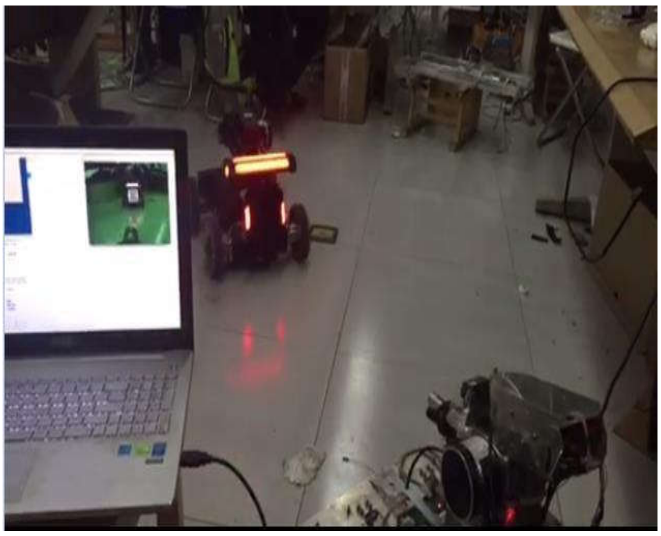


Figure 3 device physical map

VII Concluding remarks

In this paper, the target recognition model of combat robot based on Opencv image processing technology is used to obtain the image of the target object through the camera, and the contour of the image is optimized by using the image binarization technique and the image processing function in Opencv. Thus, the robot through the camera will have a "visual", through communication with the lower computer can achieve the target combat, the experiment and the 2016 RM contest proved that this model has a certain feasibility.

References:

[1]. Robert Laganiere. Opencv computer vision programming Raiders. People Post Press, 2015
[2]. Xing Yun Mao,Fei Xue Leng.Opencv3 programming entry.Electronic Industry Press , 2015
[3]. Yu Jing Zhang.Computer Vision Tutorial. People Post Press,2013
[4]. JohnBlankenship. Robot Programming Technology and Realization. Science Press,2011