转 OpenCV 使用光流法检测物体运动

OpenCV 可以使用光流法检测物体运动，贴上代码以及效果。

```cpp
[cpp]
1.    // opticalflow.cpp : 定义控制台应用程序的入口点。
2.    //
3.
4.    #include "stdafx.h"
5.
6.
7.    // Example 10-1. Pyramid Lucas-Kanade optical flow code
8.    //
9.    /* *************** License:*************************
10.      Oct. 3, 2008
11.      Right to use this code in any way you want without warrenty, support or any guarentee of it working.
12.
13.      BOOK: It would be nice if you cited it:
14.      Learning OpenCV: Computer Vision with the OpenCV Library
15.        by Gary Bradski and Adrian Kaehler
16.        Published by O'Reilly Media, October 3, 2008
17.
18.      AVAILABLE AT:
19.        http://www.amazon.com/Learning-OpenCV-Computer-Vision-Library/dp/0596516134
20.        Or: http://oreilly.com/catalog/9780596516130/
21.        ISBN-10: 0596516134 or: ISBN-13: 978-0596516130
22.
23.      OTHER OPENCV SITES:
24.      * The source code is on sourceforge at:
25.        http://sourceforge.net/projects/opencvlibrary/
26.      * The OpenCV wiki page (As of Oct 1, 2008 this is down for changing over servers, but should come back):
27.        http://opencvlibrary.sourceforge.net/
28.      * An active user group is at:
29.        http://tech.groups.yahoo.com/group/OpenCV/
30.      * The minutes of weekly OpenCV development meetings are at:
31.        http://pr.willowgarage.com/wiki/OpenCV
32.      ********************************************** */
33.
34.    #include <cv.h>
35.    #include <cxcore.h>
36.    #include <highgui.h>
37.    #include <stdio.h>
38.
39.    const int MAX_CORNERS = 500;
40.    int main(int argc, char** argv) {
41.       // Initialize, load two images from the file system, and
42.       // allocate the images and other structures we will need for
43.       // results.
44.       //
45.       IplImage* imgA = cvLoadImage("OpticalFlow0.jpg",CV_LOAD_IMAGE_GRAYSCALE);
46.       IplImage* imgB = cvLoadImage("OpticalFlow1.jpg",CV_LOAD_IMAGE_GRAYSCALE);
47.       CvSize      img_sz    = cvGetSize( imgA );
48.       int         win_size = 10;
49.       IplImage* imgC = cvLoadImage("OpticalFlow1.jpg",CV_LOAD_IMAGE_UNCHANGED);
50.
51.       // The first thing we need to do is get the features
52.       // we want to track.
53.       //
54.       IplImage* eig_image = cvCreateImage( img_sz, IPL_DEPTH_32F, 1 );
55.       IplImage* tmp_image = cvCreateImage( img_sz, IPL_DEPTH_32F, 1 );
56.       int            corner_count = MAX_CORNERS;
57.       CvPoint2D32f* cornersA     = new CvPoint2D32f[ MAX_CORNERS ];
58.       cvGoodFeaturesToTrack(
59.          imgA,
60.          eig_image,
61.          tmp_image,
62.          cornersA,
63.          &corner_count,
64.          0.01,
65.          5.0,
66.          0,
67.          3,
68.          0,
69.          0.04
70.       );
71.       cvFindCornerSubPix(
72.          imgA,
73.          cornersA,
74.          corner_count,
75.          cvSize(win_size,win_size),
76.          cvSize(-1,-1),
77.          cvTermCriteria(CV_TERMCRIT_ITER|CV_TERMCRIT_EPS,20,0.03)
78.       );
79.       // Call the Lucas Kanade algorithm
80.       //
```

```
81.        char features_found[ MAX_CORNERS ];
82.        float feature_errors[ MAX_CORNERS ];
83.     CvSize pyr_sz = cvSize( imgA->width+8, imgB->height/3 );
84.     IplImage* pyrA = cvCreateImage( pyr_sz, IPL_DEPTH_32F, 1 );
85.   IplImage* pyrB = cvCreateImage( pyr_sz, IPL_DEPTH_32F, 1 );
86.   CvPoint2D32f* cornersB        = new CvPoint2D32f[ MAX_CORNERS ];
87.   cvCalcOpticalFlowPyrLK(
88.        imgA,
89.        imgB,
90.        pyrA,
91.        pyrB,
92.        cornersA,
93.        cornersB,
94.        corner_count,
95.        cvSize( win_size,win_size ),
96.        5,
97.        features_found,
98.        feature_errors,
99.        cvTermCriteria( CV_TERMCRIT_ITER | CV_TERMCRIT_EPS, 20, .3 ),
100.       0
101.   );
102.   // Now make some image of what we are looking at:
103.   //
104.   for( int i=0; i<corner_count; i++ ) {
105.       if( features_found[i]==0|| feature_errors[i]>550 ) {
106. //        printf("Error is %f/n",feature_errors[i]);
107.           continue;
108.       }
109. //    printf("Got it/n");
110.       CvPoint p0 = cvPoint(
111.           cvRound( cornersA[i].x ),
112.           cvRound( cornersA[i].y )
113.       );
114.       CvPoint p1 = cvPoint(
115.           cvRound( cornersB[i].x ),
116.           cvRound( cornersB[i].y )
117.       );
118.       cvLine( imgC, p0, p1, CV_RGB(255,0,0),2 );
119.   }
120.   cvNamedWindow("ImageA",0);
121.   cvNamedWindow("ImageB",0);
122.   cvNamedWindow("LKpyr_OpticalFlow",0);
123.   cvShowImage("ImageA",imgA);
124.   cvShowImage("ImageB",imgB);
125.   cvShowImage("LKpyr_OpticalFlow",imgC);
126.   cvWaitKey(0);
127.   return 0;
128. }
```
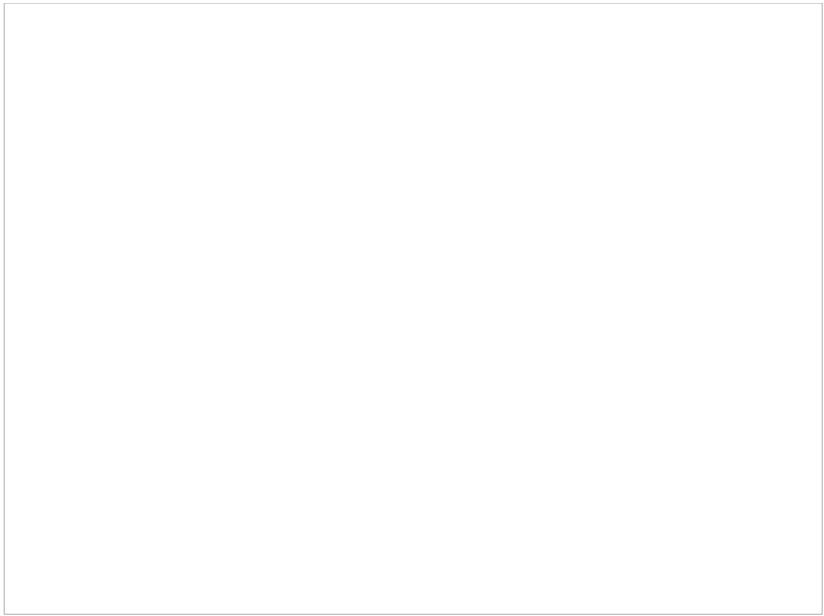
两张测试图片：

OpticalFlow0.jpg
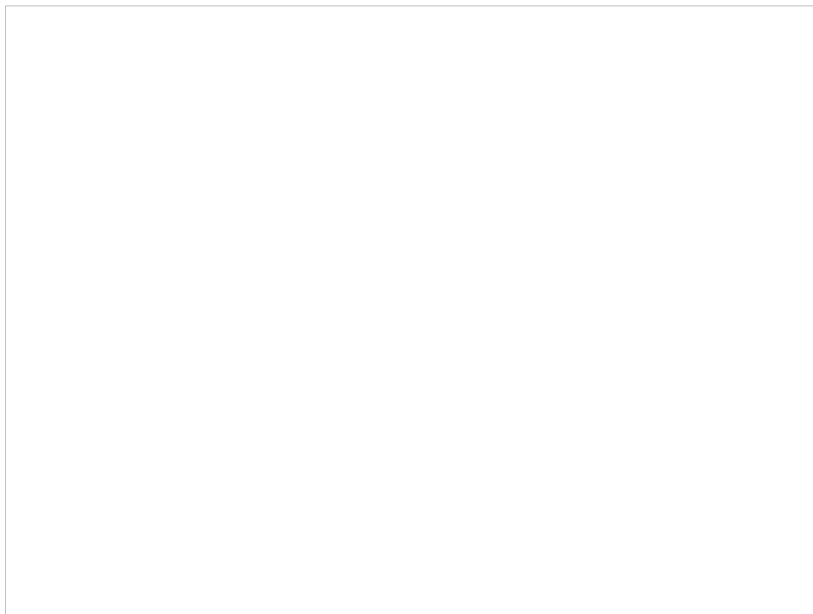
OpticalFlow1.jpg

运行结果：

文章标签： OpenCV 光流法 运动检测 源代码

个人分类： OpenCV