OpenCV 使用光流法检测物体运动

2013年11月10日 16:42:49 阅读数:17024

OpenCV 可以使用光流法检测物体运动,贴上代码以及效果。

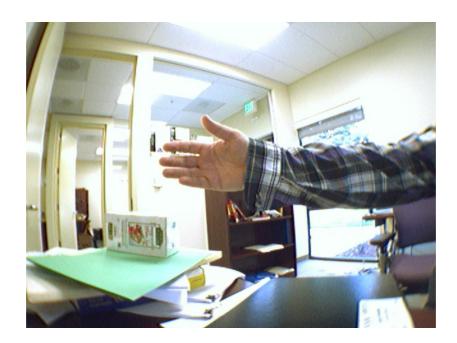
```
[cpp] 📳 📑
      // opticalflow.cpp : 定义控制台应用程序的入口点。
1.
      //
2.
3.
4.
     #include "stdafx.h"
5.
6.
7.
      // Example 10-1. Pyramid Lucas-Kanade optical flow code
8.
     9.
      Oct. 3, 2008
10.
11.
         Right to use this code in any way you want without warrenty, support or any guarentee of it working.
12.
13.
         BOOK: It would be nice if you cited it:
      Learning OpenCV: Computer Vision with the OpenCV Library
14.
15.
           by Gary Bradski and Adrian Kaehler
     Published by O'Reilly Media, October 3, 2008
16.
17.
18.
     AVAILABLE AT:
          http://www.amazon.com/Learning-OpenCV-Computer-Vision-Library/dp/0596516134
19.
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.vahoo.com/group/OpenCV/
         * The minutes of weekly OpenCV development meetings are at:
30.
          http://pr.willowgarage.com/wiki/OpenCV
31.
32.
33.
34.
      #include <cv.h>
35.
      #include <cxcore.h>
36.
      #include <highgui.h>
37.
      #include <stdio.h>
38.
      const int MAX CORNERS = 500;
39.
      int main(int argc, char** argv) {
40.
41.
         // Initialize, load two images from the file system, and
42.
        // allocate the images and other structures we will need for
43.
         // results.
      //
44.
         IplImage* imgA = cvLoadImage("OpticalFlow0.jpg",CV_LOAD_IMAGE_GRAYSCALE);
45.
         IplImage* imgB = cvLoadImage("OpticalFlow1.jpg",CV_LOAD_IMAGE_GRAYSCALE);
46.
                  img_sz = cvGetSize( imgA );
win_size = 10;
47.
          CvSize
48.
      int
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 );
          IplImage* tmp_image = cvCreateImage( img_sz, IPL_DEPTH_32F, 1 );
55.
                       corner_count = MAX_CORNERS;
56.
         int
          CvPoint2D32f* cornersA
57.
                                       = new CvPoint2D32f[ MAX CORNERS ];
         cvGoodFeaturesToTrack(
58.
             imgA,
59.
             eig_image,
60.
              tmp_image,
61.
62.
             cornersA.
63.
              &corner_count,
64.
             0.01.
65.
              5.0,
66.
             Θ,
67.
              3,
              Θ,
68.
69.
              0.04
      );
70.
71.
          cvFindCornerSubPix(
72.
             imaA.
              cornersA,
73.
             corner count.
74.
75.
              cvSize(win_size,win_size),
76.
             cvSize(-1.-1).
              cvTermCriteria(CV_TERMCRIT_ITER|CV_TERMCRIT_EPS,20,0.03)
77.
78.
79.
          // Call the Lucas Kanade algorithm
80.
```

```
81.
            char features_found[ MAX_CORNERS ];
 82.
           float feature errors[ MAX CORNERS ];
            CvSize pyr sz = cvSize( imgA->width+8, imgB->height/3 );
 83.
 84.
           IplImage* pyrA = cvCreateImage( pyr_sz, IPL_DEPTH_32F, 1 );
         IplImage* pyrB = cvCreateImage( pyr_sz, IPL_DEPTH_32F, 1 );
 85.
         CvPoint2D32f* cornersB
 86.
                                       = new CvPoint2D32f[ MAX_CORNERS ];
         cvCalcOpticalFlowPyrLK(
 87.
 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,
            feature_errors,
 98.
 99.
            cvTermCriteria( CV_TERMCRIT_ITER | CV_TERMCRIT_EPS, 20, .3 ),
100.
101.
        // Now make some image of what we are looking at:
102.
103.
104.
         for( int i=0; i<corner_count; i++ ) {</pre>
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");
        //
            CvPoint p0 = cvPoint(
110.
111.
               cvRound( cornersA[i].x ),
112.
               cvRound( cornersA[i].y )
113.
114.
            CvPoint p1 = cvPoint(
               cvRound( cornersB[i].x ),
115.
               cvRound( cornersB[i].y )
116.
117.
            );
            cvLine( imgC, p0, p1, CV_RGB(255,0,0),2
118.
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.
```

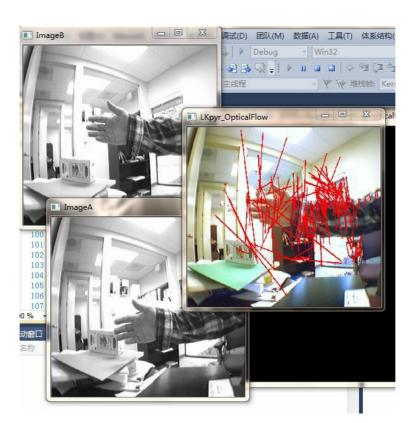
两张测试图片:

OpticalFlow0.jpg





运行结果:



文章标签: OpenCV 光流法 运动检测 源代码 个人分类: OpenCV

此PDF由spygg生成,请尊重原作者版权!!!

我的邮箱:liushidc@163.com