# Camshift算法原理及其Opencv实现

2013年10月02日 15:31:06 阅读数:33718

#### Camshift原理

camshift利用目标的颜色直方图模型将图像转换为颜色概率分布图,初始化一个搜索窗的大小和位置,并根据上一帧得到的结果自适应调整搜索窗口的位置和大小,从而定位出当前图像中目标的中心位置。

## 分为三个部分:

# 1--色彩投影图(反向投影):

(1).RGB颜色空间对光照亮度变化较为敏感,为了减少此变化对跟踪效果的影响,首先将图像从RGB空间转换到HSV空间。(2).然后对其中的H分量作直方图,在直方图中代表了不同H分量值出现的概率或者像素个数,就是说可以查找出H分量大小为h的概率或者像素个数,即得到了颜色概率查找表。(3).将图像中每个像素的值用其颜色出现的概率对替换,就得到了颜色概率分布图。这个过程就叫反向投影,颜色概率分布图是一个灰度图像。

#### 2--meanshift

meanshift算法是一种密度函数梯度估计的非参数方法,通过迭代寻优找到概率分布的极值来定位目标。

#### 算法过程为:

- (1).在颜色概率分布图中选取搜索窗W
- (2).计算零阶距:

$$M_{00} = \sum_{x} \sum_{y} I(x, y)$$

计算一阶距

$$M_{10} = \sum_{x} \sum_{y} x I(x, y); M_{01} = \sum_{x} \sum_{y} y I(x, y)$$

计算搜索窗的质心:

$$x_c = M_{10} / M_{00}; y_c = M_{01} / M_{00}$$

(3).调整搜索窗大小

$$s = \sqrt{M_{00} / 256}$$
; Kg为1.2s;

(4).移动搜索窗的中心到质心,如果移动距离大于预设的固定阈值,则重复2)3)4),直到搜索窗的中心与质心间的移动距离小于预设的固定阈值,或 者循环运算的次数达到某一最大值,停止计算。关于meanshift的收敛性证明可以google相关文献。

## 3--camshift

将meanshift算法扩展到连续图像序列,就是camshift算法。它将视频的所有帧做meanshift运算,并将上一帧的结果,即搜索窗的大小和中心,作为下一帧meanshift算法搜索窗的初始值。如此迭代下去,就可以实现对目标的跟踪。

## 算法过程为:

- (1).初始化搜索窗
- (2).计算搜索窗的颜色概率分布(反向投影)
- (3).运行meanshift算法,获得搜索窗新的大小和位置。
- (4).在下一帧视频图像中用(3)中的值重新初始化搜索窗的大小和位置,再跳转到(2)继续进行。

camshift能有效解决目标变形和遮挡的问题,对系统资源要求不高,时间复杂度低,在简单背景下能够取得良好的跟踪效果。但当背景较为复杂,或者有许多与目标颜色相似像素干扰的情况下,会导致跟踪失败。因为它单纯的考虑颜色直方图,忽略了目标的空间分布特性,所以这种情况下需加入对跟踪目标的预测算法。

OpenCV实现camshift算法的例子:

```
[cpp] 📳 📑
1.
    //对运动物体的跟踪:
    //如果背景固定,可用帧差法 然后在计算下连通域 将面积小的去掉即可
2.
3.
    //如果背景单一,即你要跟踪的物体颜色和背景色有较大区别 可用基于颜色的跟踪 如CAMSHIFT 鲁棒性都是较好的
    //如果背景复杂,如背景中有和前景一样的颜色 就需要用到一些具有预测性的算法 如卡尔曼滤波等 可以和CAMSHIFT结合
4.
5.
    #ifdef CH
6.
7.
    #pragma package <opencv>
8.
    #endif
9.
10.
    #ifndef _EiC
11.
    #include "cv.h"
    #include "highgui.h"
    #include <stdio.h>
```

```
#IIICLUUE <CLYPE.II>
 15.
       #endif
16.
      IplImage *image = 0, *hsv = 0, *hue = 0, *mask = 0, *backproject = 0, *histimg = 0;
 17.
      //用HSV中的Hue分量进行跟踪
18.
 19.
      CvHistogram *hist = 0;
20.
       //直方图类
 21.
       int backproject_mode = 0;
22.
      int select_object = 0;
 23.
      int track object = 0;
 24.
      int show_hist = 1;
 25.
       CvPoint origin;
 26.
      CvRect selection;
 27.
      CvRect track window;
 28.
      CvBox2D track_box;
      //Meanshift跟踪算法返回的Box类
 29.
      //typedef struct CvBox2D{
 30.
      //CvPoint2D32f center; /* 盒子的中心 */
//CvSize2D32f size; /* 盒子的长和宽 */
31.
 32.
       //float angle; /* 水平轴与第一个边的夹角,用弧度表示*/
 33.
 34.
       //}CvBox2D:
 35.
      CvConnectedComp track_comp;
 36.
       //连接部件
 37.
       //typedef struct CvConnectedComp{
 38.
      //double area; /* 连通域的面积 */
 39.
       //float value; /* 分割域的灰度缩放值 */
      //CvRect rect; /* 分割域的 ROI */
 40.
 41.
       //} CvConnectedComp;
42.
      int hdims = 16;
      //划分直方图bins的个数,越多越精确
43.
44.
       float hranges_arr[] = {0,180};
       //像素值的范围
45.
46.
       float* hranges = hranges_arr;
47.
       //用于初始化CvHistogram类
48.
      int vmin = 10, vmax = 256, smin = 30;
49.
       //用干设置滑动条
50.
 51.
      {f void} on_mouse( int event, int x, int y, int flags, {f void}^* param )
 52.
      //鼠标回调函数,该函数用鼠标进行跟踪目标的选择
 53.
 54.
           if( !image )
 55.
              return;
 56.
 57.
          if( image->origin )
 58.
              y = image->height - y;
           //如果图像原点坐标在左下,则将其改为左上
 59.
60.
61.
          if( select object )
       _____//select_object为1,表示在用鼠标进行目标选择
 62.
63.
           //此时对矩形类selection用当前的鼠标位置进行设置
 64.
 65.
              selection.x = MIN(x,origin.x);
 66.
              selection.y = MIN(y,origin.y);
 67.
              selection.width = selection.x + CV_IABS(x - origin.x);
 68.
              selection.height = selection.y + CV_IABS(y - origin.y);
 69.
 70.
              selection.x = MAX( selection.x, 0 );
 71.
              selection.y = MAX( selection.y, 0 );
              selection.width = MIN( selection.width, image->width );
 72.
              selection.height = MIN( selection.height, image->height );
 73.
              selection.width -= selection.x:
 74.
 75.
              selection.height -= selection.y;
 76.
 77.
 78.
      switch( event )
 79.
 80.
          case CV_EVENT_LBUTTONDOWN:
 81.
              //鼠标按下,开始点击选择跟踪物体
 82.
              origin = cvPoint(x,y);
 83.
              selection = cvRect(x,y,0,0);
 84.
              select_object = 1;
 85.
              break;
           case CV_EVENT_LBUTTONUP:
86.
87.
              //鼠标松开,完成选择跟踪物体
88.
              select object = 0:
              if( selection.width > 0 && selection.height > 0 )
89.
              //如果选择物体有效,则打开跟踪功能
90.
91.
                  track object = -1;
92.
              break:
 93.
 94.
 95.
 96.
 97.
       CvScalar hsv2rgb( float hue )
 98.
      //用于将Hue量转换成RGB量
99.
100.
          int rgb[3], p, sector;
          static const int sector data[][3]=
101.
            \{\{0,2,1\}, \{1,2,0\}, \{1,0,2\}, \{2,0,1\}, \{2,1,0\}, \{0,1,2\}\};
102.
          103.
          sector = cvFloor(hue);
104.
          n = cvRound(255*(hue - sector)):
```

```
106.
        p ^= sector & 1 ? 255 : 0;
107.
108.
       rgb[sector_data[sector][0]] = 255;
109.
           rgb[sector_data[sector][1]] = 0;
110.
           rgb[sector_data[sector][2]] = p;
111.
112.
          return cvScalar(rgb[2], rgb[1], rgb[0],0);
113.
       }
114.
115.
       int main( int argc, char** argv )
116.
       {
117.
           CvCapture* capture = 0:
118.
119.
           if( argc == 1 || (argc == 2 && strlen(argv[1]) == 1 && isdigit(argv[1][0])))
120.
               //打开摄像头
121.
               capture = cvCaptureFromCAM(argc == 2 ? argv[1][0] - '0' : 0);
           else if( argc == 2 )
122.
123.
               //打开avi
124.
               capture = cvCaptureFromAVI( argv[1] );
125.
       if( !capture )
126.
127.
           //打开视频流失败
       {
128.
129.
                fprintf(stderr, "Could not initialize capturing...\n");
130.
               return -1;
131.
           }
132.
           printf( "Hot keys: \n"
133.
134.
              "\tESC - quit the program\n"
135.
                "\tc - stop the tracking\n"
136.
               "\tb - switch to/from backprojection view\n"
137.
                "\th - show/hide object histogram\n'
               "To initialize tracking, select the object with mouse\n" );
138.
139.
           //打印程序功能列表
140.
141.
           cvNamedWindow( "Histogram". 1 ):
          //用于显示直方图
142.
           cvNamedWindow( "CamShiftDemo", 1 );
143.
           //用于显示视频
144.
145.
           cvSetMouseCallback( "CamShiftDemo", on mouse, 0 );
146.
           //设置鼠标回调函数
           cvCreateTrackbar( "Vmin", "CamShiftDemo", &vmin, 256, 0 );
cvCreateTrackbar( "Vmax", "CamShiftDemo", &vmax, 256, 0 );
147.
148.
149.
           cvCreateTrackbar( "Smin", "CamShiftDemo", &smin, 256, 0 );
       //设置滑动条
150.
151.
152.
        for(;;)
153.
           //进入视频帧处理主循环
154.
155.
               IplImage* frame = 0;
156.
              int i, bin w, c;
157.
158.
               frame = cvQueryFrame( capture );
159.
               if( !frame )
160.
                  break:
161.
162.
               if( !image )
163.
               //image为0,表明刚开始还未对image操作过,先建立一些缓冲区
164.
                   image = cvCreateImage( cvGetSize(frame), 8, 3 );
165.
166.
                   image->origin = frame->origin;
167.
                   hsv = cvCreateImage( cvGetSize(frame), 8, 3 );
168.
                   hue = cvCreateImage( cvGetSize(frame), 8, 1 );
169.
                   mask = cvCreateImage( cvGetSize(frame), 8, 1 );
170.
                   //分配掩膜图像空间
171.
                   backproject = cvCreateImage( cvGetSize(frame), 8, 1 );
172.
                   //分配反向投影图空间,大小一样,单通道
                   \label{eq:hist_array}  \mbox{hist = cvCreateHist( 1, \&hdims, CV\_HIST\_ARRAY, \&hranges, 1 );} 
173.
174.
                   //分配直方图空间
175.
                   histimg = cvCreateImage( cvSize(320,200), 8, 3 );
176
                   //分配用于直方图显示的空间
177.
                   cvZero( histimg );
178.
                   //置背景为黑色
179.
               }
180.
181.
               cvCopy(frame, image, 0);
182.
               cvCvtColor( image, hsv, CV_BGR2HSV );
               //把图像从RGB表色系转为HSV表色系
183.
184.
185.
               if( track_object )
               //track object非零,表示有需要跟踪的物体
186.
187.
               {
188.
                   int vmin = vmin, vmax = vmax;
189.
190.
                   cvInRangeS( hsv, cvScalar(0,smin,MIN(_vmin,_vmax),0),
191.
                               cvScalar(180,256,MAX(\_vmin,\_vmax),0), mask);
                   //制作掩膜板,只处理像素值为H:0~180,S:smin~256,V:vmin~vmax之间的部分
192.
193.
                   cvSplit( hsv, hue, 0, 0, 0 );
                   //分离H分量
194.
195.
196
                   if( track object < 0 )</pre>
```

```
197.
                   //如果需要跟踪的物体还没有进行属性提取,则进行选取框类的图像属性提取
198.
199.
                       float max val = 0.f:
                       cvSetImageR0I( hue, selection );
200.
201.
                       //设置原选择框为ROI
202.
                       cvSetImageR0I( mask, selection );
                       //设置掩膜板选择框为R0I
203
204.
                       cvCalcHist( &hue, hist, 0, mask );
                       //得到选择框内且满足掩膜板内的直方图
205.
206.
                       cvGetMinMaxHistValue( hist, 0, &max_val, 0, 0 );
207.
                       cvConvertScale( hist->bins, hist->bins, max_val ? 255. / max_val : 0., 0 );
208.
                       // 对直方图的数值转为0~255
209.
                       cvResetImageR0I( hue );
210.
                       //去除R0I
211.
                       cvResetImageR0I( mask );
                       //去除R0I
212.
213.
                       track window = selection;
                       track object = 1;
214.
215.
                       //置track object为1,表明属性提取完成
216.
                       cvZero( histimg );
217.
                       bin_w = histimg->width / hdims;
218.
                       for( i = 0; i < hdims; i++ )</pre>
219.
                       //画直方图到图像空间
220.
221.
                           int val = cvRound( cvGetReal1D(hist->bins,i)*histimg->height/255 );
222.
                          CvScalar color = hsv2rgb(i*180.f/hdims);
223.
                           cvRectangle( histimg, cvPoint(i*bin_w,histimg->height),
224.
                                      cvPoint((i+1)*bin_w,histimg->height - val),
                                       color, -1, 8, 0 );
225.
226.
227.
228.
229.
                   cvCalcBackProject( &hue, backproject, hist );
230
                   //计算hue的反向投影图
231.
                   cvAnd( backproject, mask, backproject, 0 );
232.
                   //得到掩膜内的反向投影
233.
                   cvCamShift( backproject, track_window,
234.
                              cvTermCriteria( CV_TERMCRIT_EPS | CV_TERMCRIT_ITER, 10, 1
235.
                              &track_comp, &track_box );
236.
                   //使用MeanShift算法对backproject中的内容进行搜索,返回跟踪结果
237.
                   track window = track comp.rect;
                   //得到跟踪结果的矩形框
238.
239.
240.
                   if( backproject mode )
241.
                       cvCvtColor( backproject, image, CV GRAY2BGR );
242.
243.
                   if( image->origin )
                      track box.angle = -track box.angle;
244.
245
                   cvEllipseBox( image, track box, CV RGB(255,0,0), 3, CV AA, 0 );
                   //画出跟踪结果的位置
246.
247
               }
248.
249.
               if( select_object && selection.width > 0 && selection.height > 0 )
250.
               //如果正处于物体选择,画出选择框
251.
               {
252.
                   cvSetImageR0I( image, selection );
253.
                   cvXorS( image, cvScalarAll(255), image, 0 );
254.
                   cvResetImageR0I( image );
255.
               }
256.
               cvShowImage( "CamShiftDemo". image ):
257.
               cvShowImage( "Histogram", histimg );
258.
259.
260.
               c = cvWaitKey(10);
261.
               if((char) c == 27)
262.
                  break;
263.
               switch( (char) c )
264.
               //按键切换功能
265.
266.
               case 'b':
267.
                   backproject_mode ^= 1;
268.
                  break;
269.
               case 'c':
                  track object = 0;
270.
271.
                   cvZero( histimg );
272.
                  break:
273.
               case 'h':
274.
                  show hist ^= 1;
275.
                   if( !show hist )
276.
                      cvDestroyWindow( "Histogram" );
277.
278.
                      cvNamedWindow( "Histogram", 1 )
279.
                   break;
280.
               default:
281.
282.
283.
284.
285.
           cvReleaseCapture( &capture );
286.
           cvDestrovWindow("CamShiftDemo"):
287.
```

