

...b\11. Image Analysis with Microcomputer\exercises\Ex3-CV-main.cpp

```
1  /*****
2  *
3  *
4  *
5  *****/
6
7  #include <opencv2/opencv.hpp>
8  #include <iostream>
9  #include "Ex3-CV-header.h"
10
11  using namespace cv;
12  using namespace std;
13
14
15  int main(int argc, char ** argv) {
16      //double t = (double)getTickCount(); // TIMING
17
18      ///_____DON'T OVERWRITE THE ORIGINAL IMAGE!!!_____
19      IplImage * original, * filtered;
20
21      /// Upload in grey scale with a resolution 640x480
22      original = cvLoadImage(argv[1] , CV_LOAD_IMAGE_GRAYSCALE);
23
24      /// To use 'lowPass'
25      filtered = cvCreateImage(cvGetSize(original) , IPL_DEPTH_8U , 1);
26      cvSet(filtered , 0); // Set to black
27
28
29
30
31  /*****
32  *
33  *
34  *
35  *****/
36
37  /*****
38  *
39  *
40  *
41  *****/
42
43  PARTS 1 & 2
44  -----
45  *
46  *
47  *
48  *****/
49
50  Low- and High-Pass Filtering
51
52  /*****
53  *
54  *
55  *
56  *****/
57
58  ///_____My version (Low-Pass)_____
59  /// Slower but it removes much more noise in uniform areas.
60  lowPass(original, filtered);
61
62  /// Using OpenCV function (Low-Pass):
63  IplImage * LP_filtered = cvCreateImage( cvGetSize(original) , IPL_DEPTH_8U , 1 );
64  IplImage * HP_filtered = cvCreateImage( cvGetSize(original) , IPL_DEPTH_8U , 1 );
65
66  ///_____Building the Kernels for the filters_____
67
68  /// LOW PASS
69
70  float LP_mat [3][3];
71
72  for(int i = 0 ; i < 3 ; i++)
73      for(int j = 0 ; j < 3 ; j++)
74          LP_mat[i][j] = 1.0 / 9;
75
76  CvMat LP_kernel = cvMat(3 , 3 , CV_32F , LP_mat);
77
78  /*****
79  *
80  *
81  *
82  *****/
83
84  /// HIGH PASS
85
86  float HP_mat [3][3] = { { 0 , -1.0/4 , 0 },
87                          { -1.0/4 , 2.0 , -1.0/4 },
88                          { 0 , -1.0/4 , 0 } };
```

```
64   CvMat HP_kernel = cvMat(3 , 3 , CV_32F , HP_mat);
65
66   /// _____ FILTERING _____
67
68   cvFilter2D(original , HP_filtered , &HP_kernel , cvPoint(-1,-1) );
69   cvFilter2D(original , LP_filtered , &LP_kernel , cvPoint(-1,-1) );
70
71
72
73   /*****
74   *                               PART 3                               *
75   *                               -----                               *
76   *                               Percentile Filter                       *
77   *****/
78
79   /// _____ First attempt - imaged cropped (rim rejected) _____
80   /// Doesn't really work that well... Destroys some details and doesn't reduce
81   /// much noise
82
83   IplImage * perc_filtered = cvCreateImage(cvSize(original->width - 2, original->height - 2),
84                                           IPL_DEPTH_8U, 1);
85
86   fractileFilter(original, perc_filtered, 50);                      // Median filter: 50%
87
88
89
90   /*****
91   *                               PART 4                               *
92   *                               -----                               *
93   *                               Laplace-Gaussian Filter                 *
94   *****/
95
96   IplImage * LG_filtered = cvCreateImage(cvGetSize(original), IPL_DEPTH_8U , 1);
97
98   /*float LG_mat [9][9] = { {0,0,1,2,2,2,1,0,0},
99                             {0,1,5,10,12,10,5,1,0},
100                             {1,5,15,19,16,19,15,5,1},
101                             {2,10,19,-19,-64,-19,19,10,2},
102                             {2,12,16,-64,-148,-64,16,12,2},
103                             {2,10,19,-19,-64,-19,19,10,2},
104                             {1,5,15,19,16,19,15,5,1},
105                             {0,1,5,10,12,10,5,1,0},
106                             {0,0,1,2,2,2,1,0,0}
107                             };
108
109   CvMat LG_kernel = cvMat(9, 9, CV_32F, LG_mat); */
110
111   /// This gives a much better result!!!
112
113   /*float LG_mat [3][3] = { { -1 , -1, -1},
114                             { -1 ,  8, -1},
115                             { -1 , -1, -1} };*/
116
117   CvMat LG_kernel = cvMat(3 , 3 , CV_32F , LG_mat);
118
119   /* NOTE: Probably because of how the function cvFilter2D is implemented it's better
120      to use a matrix of float and to use a constructor of CvMat with CV_32F.
121      Even when the values are int, by specifying e.g. CV_16S, results are
122      weird and more processing is necessary to obtain the right visual result...*/
123
124
125
126
```

```
127  /// Filtering
128
129  cvFilter2D(original , LG_filtered , &LG_kernel , cvPoint(-1,-1) );
130
131
132  //*****
133
134  ///_____Stream from camera in normal and filtered mode (2 separate windows)_____
135
136  doubleStream(&LG_kernel);
137
138
139
140
141  /*****
142  *                               PARTS 5                               *
143  *                               -----                               *
144  *                               Triangular Filter                               *
145  *****/
146
147  IplImage * x_filtered = cvCreateImage(cvGetSize(original) , IPL_DEPTH_8U , 1);
148  IplImage * y_filtered = cvCreateImage(cvGetSize(original) , IPL_DEPTH_8U , 1);
149
150  /// Emphasize the horizontal direction gradients
151
152  xGradientFilter(original, x_filtered);
153
154  /// Emphasize the vertical direction gradients
155
156  yGradientFilter(original, y_filtered);
157
158
159
160
161  /*****
162  *                               Display and Save                               *
163  *****/
164
165  cvNamedWindow("Low-Pass Filtered Image (mine)" , WINDOW_NORMAL);
166  cvNamedWindow("Original Image (grey scale)" , WINDOW_NORMAL);
167  cvNamedWindow("Laplace-Gauss Filtered Image" , WINDOW_NORMAL);
168  cvNamedWindow("Low-Pass Filtered Image" , WINDOW_NORMAL);
169  cvNamedWindow("High-Pass Filtered Image" , WINDOW_NORMAL);
170  cvNamedWindow("Fractile Filtered Image" , WINDOW_NORMAL);
171  cvNamedWindow("x gradient" , WINDOW_NORMAL);
172  cvNamedWindow("y gradient" , WINDOW_NORMAL);
173
174  cvShowImage("Laplace-Gauss Filtered Image" , LG_filtered);
175  cvShowImage("Original Image (grey scale)" , original);
176  cvShowImage("Low-Pass Filtered Image (mine)" , filtered);
177  cvShowImage("Low-Pass Filtered Image" , LP_filtered);
178  cvShowImage("High-Pass Filtered Image" , HP_filtered);
179  cvShowImage("Fractile Filtered Image" , perc_filtered);
180  cvShowImage("x gradient" , x_filtered);
181  cvShowImage("y gradient" , y_filtered);
182
183  cvSaveImage("Original (grey scale).png" , original);
184  cvSaveImage("Laplace-Gauss_Filter.png" , LG_filtered);
185  cvSaveImage("Fractile_Filter.png" , perc_filtered);
186  cvSaveImage("My_LP_Filtered.png" , filtered);
187  cvSaveImage("LP_Filtered.png" , LP_filtered);
188  cvSaveImage("HP_Filtered.png" , HP_filtered);
189  cvSaveImage("x_gradient.png" , x_filtered);
```

```
190     cvSaveImage("y_gradient.png" , y_filtered);
191
192     cvWaitKey(0);
193
194
195     /*****
196     *                               *
197     *****/
198
199     //t = ((double)getTickCount() - t)/getTickFrequency();
200     //printf("\nExecution time = %.3f s", t);           // END TIMING
201
202     cvReleaseImage(&original);
203     cvReleaseImage(&filtered);
204     cvReleaseImage(&LP_filtered);
205     cvReleaseImage(&HP_filtered);
206     cvReleaseImage(&perc_filtered);
207     cvReleaseImage(&LG_filtered);
208     cvReleaseImage(&x_filtered);
209     cvReleaseImage(&y_filtered);
210
211     cvDestroyAllWindows();
212
213     return 0;
214 }
```