

机器视觉均值滤波作业

SZ170410221-朱方程

Code

```
1 //运行平台Ubuntu16.04
2 #include <stdlib.h>
3 #include <cv.h>
4 #include <highgui.h>
5 #include <opencv2/opencv.hpp>
6 #include <opencv2/core/core.hpp>
7 #include "ros/ros.h"
8 #include "std_msgs/String.h"
9 #include "std_msgs/Bool.h"
10 #include "std_msgs/Float32.h"
11 #include <geometry_msgs/Twist.h>
12 #include "sensor_msgs/Image.h"
13 #define LINEAR_X 0
14 using namespace cv;
15 using namespace std;
16 //按定义对图像进行均值滤波
17 void Mean(Mat input, Mat output, int MaskSize)
18 {
19     int center_k=MaskSize/2;
20     int center_l=MaskSize/2;
21     double mask = 1.00000/(MaskSize*MaskSize);
22     //将模板函数与图像进行卷积
23     for(int i=0;i<input.rows;i++){
24         for(int j=0;j<input.cols;j++){
25             double sum = 0;
26             for (int k = 0; k <MaskSize; k++){
27                 for (int l = 0; l < MaskSize; l++){
28                     if(i+(k-center_k)>=0 && j+(l-center_l)>=0)
29                         sum = sum+input.at<uchar>(i+(k-center_k),j+(l-
30 center_l))*mask;
31                 }
32             }
33             //对输出图像重新赋值
34             output.at<uchar>(i,j)=sum;
35         }
36     }
37     //可分离的滤波器
38     void Mean_Separable(Mat input, Mat output, int MaskSize)
39     {
40         int center_k=MaskSize/2;
41         int center_l=MaskSize/2;
42         double mask_row = 1.000000/MaskSize;
43         double mask_col = 1.000000/MaskSize;
44         //先将模板函数与图像进行横向卷积
45         for(int i=0;i<input.rows;i++){
```

```

46         for(int j=0;j<input.cols;j++){
47             double sum = 0;
48             for (int l = 0; l < MaskSize; l++){
49                 if(j+(l-center_l)>=0)
50                     sum = sum+input.at<uchar>(i,j+(l-center_l))*mask_row;
51             }
52             //对输出图像重新赋值
53             output.at<uchar>(i,j)=sum;
54         }
55     }
56     //先将模板函数与图像进行纵向卷积
57     for(int i=0;i<input.rows;i++){
58         for(int j=0;j<input.cols;j++){
59             double sum = 0;
60             for (int k = 0; k < MaskSize; k++){
61                 if(i+(k-center_k)>=0)
62                     sum = sum+output.at<uchar>(i+(k-center_k),j)*mask_col;
63             }
64             //对输出图像重新赋值
65             output.at<uchar>(i,j)=sum;
66         }
67     }
68 }
69
70
71 int main(int argc, char **argv)
72 {
73
74     VideoCapture capture;
75     capture.open(0); //打开 zed 相机
76
77     ROS_WARN("*****START");
78     ros::init(argc,argv,"trafficLaneTrack");
79     ros::NodeHandle n;
80
81     ros::Rate loop_rate(10);
82     ros::Publisher pub = n.advertise<geometry_msgs::Twist>
83     ("/smoother_cmd_vel", 5);
84     if (!capture.isOpened())
85     {
86         printf("摄像头没有正常打开\n");
87         return 0;
88     }
89     waitKey(1000);
90     Mat frame;
91     while (ros::ok())
92     {
93
94         Mat frIn1 = imread("/home/fangcheng/Library/lena1.png",1);
95         if(frIn1.empty())
96         {
97             break;
98         }
99         Mat frIn;
100         cvtColor(frIn1, frIn, CV_RGB2GRAY); //RGB彩色图转换成Gray灰度图
101
102

```

```

103     /*capture.read(frame);
104     if(frame.empty())
105     {
106         break;
107     }
108     Mat frIn1 = frame(cv::Rect(0, 0, frame.cols, frame.rows)); //截取
zed 的图片
109     Mat frIn;
110     cvtColor(frIn1, frIn, CV_RGB2GRAY); //RGB彩色图转换成Gray灰度
111     */
112     clock_t start, finish;
113
114     //均值滤波处理
115     Mat frMean3 = frIn.clone();
116     Mat frMean_Separable3 = frIn.clone();
117     Mat frMean5 = frIn.clone();
118     Mat frMean_Separable5 = frIn.clone();
119     //计时
120     start=clock();
121     Mean(frIn, frMean3, 3);
122     finish=clock();
123     printf("Time-consuming of 3x3 mean filter%fs\n", (double)(finish-
start)/CLOCKS_PER_SEC);
124
125     start=clock();
126     Mean_Separable(frIn, frMean_Separable3, 3);
127     finish=clock();
128     printf("Time-consuming of 3x3 separable mean filter%fs\n", (double)
(finish-start)/CLOCKS_PER_SEC);
129
130     start=clock();
131     Mean(frIn, frMean5, 5);
132     finish=clock();
133     printf("Time-consuming of 5x5 mean filter%fs\n", (double)(finish-
start)/CLOCKS_PER_SEC);
134
135     start=clock();
136     Mean_Separable(frIn, frMean_Separable5, 5);
137     finish=clock();
138     printf("Time-consuming of 5x5 separable mean filter%fs\n", (double)
(finish-start)/CLOCKS_PER_SEC);
139
140     imshow("frIn", frIn); //灰度图像
141     imshow("frMean 3x3", frMean3); //按定义编写的均值滤波
142     imshow("frMean_Separable 3x3", frMean_Separable3); //可分离的均值滤波器
143     imshow("frMean 5x5", frMean5); //按定义编写的均值滤波
144     imshow("frMean_Separable 5x5", frMean_Separable5); //可分离的均值滤波器
145     //opencv库函数均值滤波
146     Mat frMean_lib3 = frIn.clone();
147     Mat frMean_lib5 = frIn.clone();
148     blur(frIn, frMean_lib3, Size(3,3), Point(-1, -1));
149     blur(frIn, frMean_lib5, Size(5,5), Point(-1, -1));
150     imshow("frMean_lib 3x3", frMean_lib3);
151     imshow("frMean_lib 5x5", frMean_lib5);
152
153     ros::spinOnce();
154     waitKey(5);
155 }

```

```
156 | return 0;  
157 | }
```

滤波效果

原图：





耗时

```
Time-consuming of 3x3 mean filter0.014310s  
Time-consuming of 3x3 separable mean filter0.008772s  
Time-consuming of 5x5 mean filter0.036834s  
Time-consuming of 5x5 separable mean filter0.014318s
```