ソフトウェア設計及び実験
第4回レポート

6119019056 山口力也

2019/05/07 日提出

1　静止画に対する画像処理

image_cv.c を参考に, 画像の左半分の領域のみ色を変換し, 表示させるプログラムを作成せよ.

1.1　実験結果 (プログラム)

以下ソースコード 1 にソースコードを示す.

ソースコード 1: 画像を左半分のみ色を変換するプログラム

```
1         /*
2          **
3          ********************************************************************************
4          **
5          ** Project : OpenCV sample project
6          ** Module : image_cv.c
7          ** Description : OpenCV による画像の読み込み(カラープレ
               ーンの置き換え)
8          **
9          ** Version : Date: Author: Comment:
10         ** 1.0 2011/09/16(金) Akinori Tsuji Creation
11         ********************************************************************************
12         **
13         */
14        /*________ INCLUDES
              ----------------------------------------------------
              */
15 #include <stdio.h>
16 #include <opencv2/imgproc/imgproc_c.h>
```

```
17 #include <opencv2/highgui/highgui_c.h>
18
19         /*________ MACROS
               ____________________________________________________________
               */
20         /*________ DEFINITIONS
               ____________________________________________________
               */
21 #define CONVERT_RGB
22
23         /*________ VARIABLES
               ____________________________________________________
               */
24         /*________ LOCAL-FUNCTION PROTOTYPES
               ______________________________________*/
25         /*________ LOCAL-FUNCTIONS
               ________________________________________________*/
26
27         /*________ MAIN-FUNCTION
               ____________________________________________
               */
28
29 int main(int argc, char **argv)
30 {
31         int x, y;
32         uchar p[3];
33         IplImage *img;
34         img = cvLoadImage("test.jpg", CV_LOAD_IMAGE_COLOR);
35         if (img == NULL) {
36                 fprintf(stderr, "*Error* cannot open test.jpg
                       \n");
37                 return -1;
38         }
39
40
41 #ifdef CONVERT_RGB
42         for (y = 0; y < img->height; y++) {
43                 for (x = 0; x < img->width; x++) {
44                         p[0] = img->imageData[img->widthStep
                               * y + x * 3]; // B
45                         p[1] = img->imageData[img->widthStep
                               * y + x * 3 + 1]; // G
46                         p[2] = img->imageData[img->widthStep
                               * y + x * 3 + 2]; // R
47
48                         // Image Processing
```

```
49                  }
50          }
51          for (y = 0; y < img->height; y++) {
52                  for (x = img->width/2; x < img->width; x++)
                        {
53
54                          img->imageData[img->widthStep * y + x
                                * 3] =
55                                 cvRound(p[0] * 1.0);
56                          img->imageData[img->widthStep * y + x
                                * 3 + 1] =
57                                 cvRound(p[1] * 1.0);
58                          img->imageData[img->widthStep * y + x
                                * 3 + 2] =
59                                 cvRound(p[2] * 1.0);
60                  }
61          }
62  #endif
63
64          cvNamedWindow("Image", CV_WINDOW_AUTOSIZE);
65          cvShowImage("Image", img);
66          cvWaitKey(0);
67
68          cvDestroyWindow("Image");
69          cvReleaseImage(&img);
70
71          return 0;
72  }
```

Image Processing の RGB を戻す部分で,for 文の条件を変更し, 画面左半分のみを描画するようにした.

1.2 実験結果 (画像)

以下図 1 に結果画像を示す.

図 1: 画面左半分のみ色を変換した結果

2 カメラを用いた動画に対するテンプレートマッチング

video_cv を参考にカメラから取得した画像におけるテンプレートマッチングを実現するプログラムを作成せよ. また, テンプレートマッチングを実施した結果について成功例, 失敗例を示せ. また, 失敗例について, なぜ失敗したのか考察せよ. なお, カラー画像に加え,2 値化画像について示せ.

2.1 実験結果

以下ソースコード 2 にソースコードを示す.

ソースコード 2: 動画に対するテンプレートマッチング

```
1  /*
2   **
3   ********************************************************************************
4   **
5   ** Project : OpenCV sample program
6   ** Module : video_cv.c
7   ** Description : OpenCV によるUSB カメラ使用
8   ** Version : Date: Author: Comment:
9   ** 1.0 2011/09/16(金) Akinori Tsuji Creation
10  ********************************************************************************
11  **
12  */
13
14 /*________ INCLUDES
       ----------------------------------------------------*/
```

```
15 #include <stdio.h>
16 #include <opencv2/core/core_c.h>
17 #include <opencv2/imgproc/imgproc_c.h>
18 #include <opencv2/highgui/highgui_c.h>
19
20 /*________ MACROS
       ------------------------------------------------------
       */
21 /*________ DEFINITIONS
       ------------------------------------------------------*/
22 #define CV_DISP_WIDTH 320 // 画面サイズ (横)
23 #define CV_DISP_HEIGHT 240 // 画面サイズ (縦)
24 #define THRESHOLD 127 // 2値化の際の閾値
25 #define THRESHOLD_MAX_VALUE 255 // 2値化の際に使用する最大値
26 #define LINE_THICKNESS 1 // 線の太さ
27 #define LINE_TYPE 8  // 線の種類
28 #define SHIFT 0   // 座標の小数点以下の桁を表すビット数
29
30
31
32 /*________ VARIABLES
       ------------------------------------------------------*/
33 /*________ LOCAL-FUNCTION PROTOTYPES
       ---------------------------------------*/
34 /*________ LOCAL-FUNCTIONS
       ------------------------------------------------*/
35
36 /*________ MAIN-FUNCTION
       ------------------------------------------------*/
37
38 int main(int argc, char **argv)
39 {
40         CvCapture *capture = 0;
41         IplImage *frame = 0;
42         double w = CV_DISP_WIDTH, h = CV_DISP_HEIGHT;
43         int ckey;
44         char windowNameTemplate[] = "Template";   // テンプレ
               ート画像を表示するウィンドウの名前
45         char windowNameBinaryTemplate[] = "BinaryTemplate";
               //2値化テンプレート画像を表示するウィンドウの名前
46         CvPoint minLocation; // 相違度が最小になる場所
47
48         // 画像を読み込む
49         IplImage *templateImage = cvLoadImage( "template.png
               ", CV_LOAD_IMAGE_ANYDEPTH | CV_LOAD_IMAGE_ANYCOLOR
                );
```

```
50
51          if ( templateImage == NULL ){
52                  // 画像が見つからなかった場合
53                  printf( "画像が見つかりません\n" );
54                  return -1;
55          }
56          // テンプレート画像をグレースケール化した画像用
                IplImage
57          IplImage *templateGrayImage = cvCreateImage( cvGetSize
                (templateImage), IPL_DEPTH_8U, 1 );
58          // テンプレート画像を 2値化した画像用IplImage
59          IplImage *templateBinaryImage = cvCreateImage(
                cvGetSize(templateImage), IPL_DEPTH_8U, 1 );
60          // BGR からグレースケールに変換する
61          cvCvtColor( templateImage, templateGrayImage,
                CV_BGR2GRAY );
62          // グレースケールから 2値に変換する
63          cvThreshold( templateGrayImage, templateBinaryImage,
                THRESHOLD, THRESHOLD_MAX_VALUE, CV_THRESH_BINARY
                );
64          // カメラのオープン (/dev/video0)
65          capture = cvCreateCameraCapture(0);
66          if (capture == NULL) {
67                  fprintf(stderr, "*Error* cannot open /dev/
                        video0\n");
68                  return -1;
69          }
70
71          // キャプチャサイズの設定
72          cvSetCaptureProperty(capture, CV_CAP_PROP_FRAME_WIDTH,
                 w);
73          cvSetCaptureProperty(capture, CV_CAP_PROP_FRAME_HEIGHT
                , h);
74          // ウィンドウを生成
75          cvNamedWindow("Capture", CV_WINDOW_AUTOSIZE);
76          cvNamedWindow("binaryCapture",CV_WINDOW_AUTOSIZE);
77          cvNamedWindow( windowNameTemplate, CV_WINDOW_AUTOSIZE
                );
78          cvNamedWindow( windowNameBinaryTemplate,
                CV_WINDOW_AUTOSIZE);
79
80          cvShowImage( windowNameTemplate,templateImage);
81          cvShowImage( windowNameBinaryTemplate,
                templateBinaryImage);
82          // カメラからフレームをキャプチャ
83          while (1) {
```

```
84              frame = cvQueryFrame(capture);
85
86              // 元画像をグレースケール化した画像用IplImage
87              IplImage *sourceGrayImage = cvCreateImage(
                    cvGetSize(frame),IPL_DEPTH_8U,1);
88              // 元画像を 2値化した画像用IplImage
89              IplImage *sourceBinaryImage = cvCreateImage(
                    cvGetSize(frame),IPL_DEPTH_8U,1);
90              // 相違度マップ画像用IplImage
91              IplImage *differenceMapImage = cvCreateImage(
                    cvSize( frame->width - templateImage->
                    width + 1, frame->height - templateImage
                    ->height + 1 ), IPL_DEPTH_32F, 1 );
92
93
94              // BGR からグレースケールに変換する
95              cvCvtColor ( frame,sourceGrayImage,CV_BGR2GRAY
                     );
96              // グレースケールから 2値に変換する
97              cvThreshold( sourceGrayImage,
                    sourceBinaryImage, THRESHOLD,
                    THRESHOLD_MAX_VALUE, CV_THRESH_BINARY );
98              // テンプレートマッチングを行う
99              cvMatchTemplate( sourceBinaryImage,
                    templateBinaryImage, differenceMapImage,
                    CV_TM_SQDIFF );
100             // テンプレートが元画像のどの部分にあるのかと
                    いう情報を得る
101             cvMinMaxLoc( differenceMapImage, NULL, NULL, &
                    minLocation, NULL, NULL );
102
103             // 一致する場所を元画像に四角で描く
104             cvRectangle(
105                             frame,
106                             minLocation,
107                             cvPoint( minLocation.x +
                                    templateImage->width,
                                    minLocation.y +
                                    templateImage->height ),
108                             CV_RGB( 255, 0, 0 ),
109                             LINE_THICKNESS,
110                             LINE_TYPE,
111                             SHIFT
112                             );
113             //キャプチャ映像の表示
114             cvShowImage("Capture", frame);
```

```
115                 // 2値化の映像を表示
116                 cvShowImage("binaryCapture",sourceBinaryImage
                        );
117
118
119                 ckey = cvWaitKey(10);
120                 cvReleaseImage(&sourceBinaryImage);
121                 cvReleaseImage(&sourceGrayImage);
122                 cvReleaseImage(&differenceMapImage);
123                 if ((ckey & 0xff) == '\x1b') {
124                         printf("Exiting ...\n");
125                         cvSaveImage("image.png",frame,0);
126                         break; // エスケープキーで終了
127                 }
128         }
129
130         //メモリを解放する
131         cvReleaseImage(&templateImage);
132         cvReleaseImage(&templateGrayImage);
133         cvReleaseImage(&templateBinaryImage);
134         cvReleaseCapture(&capture);
135
136         //window を閉じる
137         cvDestroyWindow("Capture");
138         cvDestroyWindow("binaryCapture");
139         cvDestroyWindow(windowNameTemplate);
140         cvDestroyWindow(windowNameBinaryTemplate);
141         return 0;
142 }
```

また, 以下図 2, 図 3, 図 4, 図 5 にテンプレートマッチング成功の一例を示す.

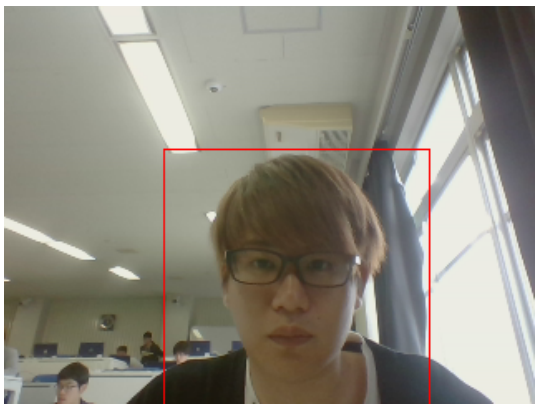

図 2: テンプレートマッチング成功の一例 (カラー)

図 3: テンプレートマッチング成功の一例 (2 値化)

図 4: テンプレートマッチング成功の一例 (カラーテンプレート)

図 5: テンプレートマッチング成功の一例 (2 値化テンプレート)

以下図 6, 図 7, 図 8, 図 9 にテンプレートマッチング失敗の一例を示す.

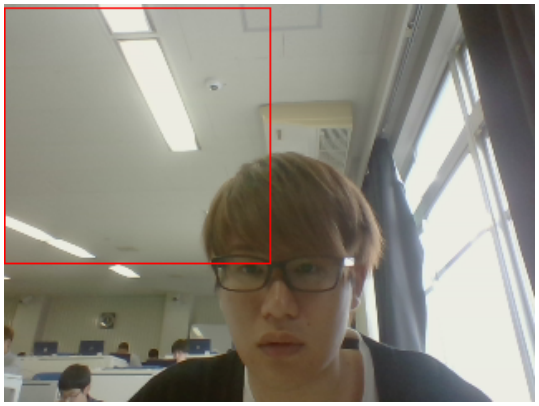

図 6: テンプレートマッチング失敗の一例 (カラー)

図 7: テンプレートマッチング失敗の一例カラ (2 値化)

図 8: テンプレートマッチング失敗の一例 (カラーテンプレート)

図 9: テンプレートマッチング失敗の一例 (2 値化テンプレート)

2.2 考察

実験から失敗するときは 2 値化する時の閾値が適切でないときであると思われる. 閾値が低すぎると 2 値化した時, 黒色となる部分が多すぎてテンプレート画像の特徴を抽出できない. 同様に閾値が高すぎると白色となる部分が多すぎてテンプレート画像の特徴を抽出できない.2 値化する時の値は 0 255 であるので中央の 127 を閾値に設定すると比較的良い精度でテンプレートマッチングをできると思われる.

3 画像の平滑化

画像の平滑化とは何か調べて述べよ. また,cvSmooth 関数を用いてテンプレート作成用画像を平滑化するプログラムを作成し, 平滑化した結果が原画像と比べてどのような画像特性となっているか述べよ.

3.1 画像の平滑化

画像の平滑化とは, 注目画素の近傍の画素の濃度値の平均を注目画素の新しい濃度値とする方法である. これにより, ごましおノイズのような高い周波数成分のノイズを除去できる. しかしながら, 雑音を多く含むような画像では完全に雑音を除去することはできない.

3.2 平滑化プログラム

以下ソースコード 3 にテンプレート作成用画像を平滑化するプログラムを示す. ここで, フィルターはガウシアンフィルタを用いた.

ソースコード 3: 動画に対するテンプレートマッチング

```
1         //インクルード
2 #include <stdio.h>
3 #include <opencv2/core/core_c.h>
4 #include <opencv2/imgproc/imgproc_c.h>
5 #include <opencv2/highgui/highgui_c.h>
6
7         int main( int argc, char **argv ) {
8
9                 char windowNameInputImage[] = "Input";
10                char windowNameOutputImage[] = "Output";
11                IplImage *inputImage,*outputImage;
12
13                //画像を読み込む
14                //入力用と出力用のIplImage を作成
15                inputImage = cvLoadImage("lena.png",
                      CV_LOAD_IMAGE_ANYDEPTH |
                      CV_LOAD_IMAGE_ANYCOLOR );
16                if ( inputImage== NULL) {
17                        // 画像が見つからなかった場合
18                        printf( "画像が見つかりません\n" );
19                        return -1;
20                }
21                outputImage =cvCloneImage(inputImage);
22
23                //平滑化フィルタをかける
24                cvSmooth( inputImage ,outputImage,CV_GAUSSIAN
                      ,1,0,0,0);
25
26                //ウィンドウを作成
27                cvNamedWindow( windowNameInputImage,
                      CV_WINDOW_AUTOSIZE );
28                cvNamedWindow( windowNameOutputImage,
                      CV_WINDOW_AUTOSIZE );
29
30                //画像を表示する
31                cvShowImage( windowNameInputImage, inputImage
                      );
32                cvShowImage( windowNameOutputImage,
                      outputImage);
33
```

```
34              // キー入力を待つ
35              cvWaitKey(0);
36
37              // 画像を保存する
38              cvSaveImage( "output.bmp", outputImage, 0 );
39              cvReleaseImage( &inputImage);
40              cvReleaseImage( &outputImage);
41              // ウィンドウを破棄する
42              cvDestroyWindow(windowNameInputImage);
43              cvDestroyWindow(windowNameOutputImage);
44
45              return 0;
46      }
```

以下図 10 に原画像を, 図 11 に結果画像を示す.

図 10: 画像の平滑化 (原画像)

図 11: 画像の平滑化 (結果画像)

3.3 平滑化の特性

原画像と比べてガウシアンフィルタをかけた時はノイズのようなものが少なくなり, 少しぼやけた画像となった. ガウシアンフィルタは注目画素からの距離に応じて近傍の画素値に重みをかける操作を行なっているため, 画素の特徴になりえる大きな値は抽出され, ノイズのような小さな値は削除されたためこのような結果になったと考えられる.

4 デバイスについて

ソフト実験で利用可能なデバイスの中から 1 つデバイスを選択し, そのデバイスを用いたゲームを考えて, 自分のアイデアをまとめよ. ゲームの概要がわかる図も入れよ.

デバイスとしては Wii リモコンを選択した.

4.1 デバイスの特徴と周辺ライブラリ

Wii リモコンは Bluetooth を使用して通信しており, そのため pc でも使うことができる. そのライブラリとして,「WiimoteLib」というライブラリが用意されている. 必要なものをダウンロードしたあと,”import WiimoteLib” とコードに書くと import して使うことができる. Wii リモコンには

- バイブレーダ
- LED
- ボタン
- 加速度センサ
- 赤外線センサ

が搭載されており, それぞれを用いてゲームなどを開発できる.

4.2 ゲーム内容

自分が作ろうと思うゲームは「イライラ棒」である. Wii リモコンでのゲーム開発は初めてであるため比較的簡単なものにした. センサを用いてリモコンを操作するだけの簡単なゲームだがセンサの制御やボタンなどを用いれるステージを用意することで搭載された機能を十分に利用できると思う. 簡易的なものではあるが以下図 12 に示すようなステージを考えた.

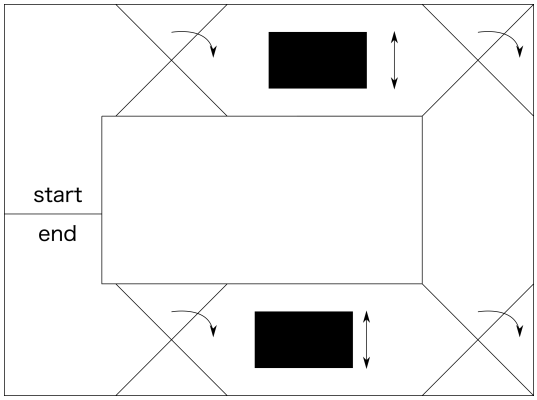

図 12: ステージの一例