画像工学レポート

No.1

1. 画像処理プログラムの基礎

・明るさ変更(bias)

・濃度反転(negate)，

・階調数変換(posterization)

2. **アナログ画像からディジタル画像への変換方法**

　 提 出 日　 2021年 5月 18日

出席番号 　IT・ES 21

　 　　 氏　 名　西川 優一郎

課題１．明るさ変換，濃度反転，階調数変換を行うプログラムを作成する。

（作成した部分のリストと元の画像と処理した画像を提出する）

課題２．アナログ画像をディジタル画像へ変換する方法を調べる。

* 注意 表紙（この頁）をつけること。

書式（フォント:9，文字数:58，行数:58）を変更しないこと。

課題内容についての考察を必ず書くこと。

インターネット参照可(ただし，ウィキペディアは不可)。

参考文献をレポートの最後に書くこと。（参照urlは必ず書くこと。）

* 提出期限： 5月　19日（水）23:59

**課題1**

本課題は授業にて配布された*mypgm\_s.h*に含まれる*load\_image\_data*関数と*save\_image\_data*関数, 変数*x\_size1*, *x\_size2*, *y\_size1*, *y\_size2*, 定数*MAX\_BRIGHTNESS*を使用している。ヘッダファイルの内容は省略する。

**画像の明るさ変更(Bias)**

Biasの変調を行うプログラムを作成した。以下にそのソースコードを示す。

|  |
| --- |
| #include<stdio.h>  #include<stdlib.h>  #include"mypgm\_s.h"  void bias\_image()  /\* image1 の明るさを変更しimage2へ出力する \*/  {  int x, y, a, wk; /\* 制御変数 \*/  printf("画素値に任意の値を加算します¥n");  printf("加算する値=");  scanf\_s("%d", &a);  x\_size2 = x\_size1; /\* 横画素数 \*/  y\_size2 = y\_size1; /\* 縦画素数 \*/  for (y = 0; y < y\_size2; y++) {  for (x = 0; x < x\_size2; x++) {  wk = a + image1[y][x];  if (wk > 255)  wk = 255;  if (wk < 0)  wk = 0;  image2[y][x] = (unsigned char)wk;  }  }  }  void main(void)  {  load\_image\_data( ); /\* 画像データを image1 に読み込む \*/  bias\_image( );  save\_image\_data( ); /\* image2 を保存する \*/  } |

ソース1　画像の明度を変更するC++プログラム

また, このプログラムの実行結果を以下に示す。

|  |  |
| --- | --- |
| 図 1 変換前画像    図 2 bias\_images関数処理後画像(明度+50)    図 3 bias\_images関数処理後画像(明度+80) | 図 4 変換前画像    図 5 bias\_images関数処理後画像(明度+80)    図 6 bias\_images関数処理後画像(明度-80) |

**画像の濃度反転処理(Negate)**

Negateの変調を行うプログラムを作成した。以下にそのソースコードを示す。

|  |
| --- |
| #include<stdio.h>  #include<stdlib.h>  #include"mypgm\_s.h"  void negate\_image()  /\* 画像の濃度を反転する \*/  {  int x, y, wk; /\* 制御変数 \*/  x\_size2 = x\_size1; /\* 横画素数 \*/  y\_size2 = y\_size1; /\* 縦画素数 \*/  for (y = 0; y < y\_size2; y++) {  for (x = 0; x < x\_size2; x++) {  wk = MAX\_BRIGHTNESS - image1[y][x];  if (wk> 255)  wk = 255;  if (wk< 0)  wk = 0;  image2[y][x] = (unsigned char)wk;  //image2[y][x] = image1[y][x] ;  }  }  }  void main(void)  {  load\_image\_data( ); /\* 画像データを image1 に読み込む \*/  negate\_image ( );  save\_image\_data( ); /\* image2 を保存する \*/  } |

ソース1　画像の明度を変更するC++プログラム

また, このプログラムの実行結果を以下に示す。

|  |  |
| --- | --- |
| 図 7 変換元画像    図 8 negate\_image関数処理後画像 | 図 9 変換元画像    図 10 negate\_image関数処理後画像 |

**画像の階調数変換処理(Posterization\_image)**

Posterization\_imageの変調を行うプログラムを作成した。以下にそのソースコードを示す。

|  |
| --- |
| #include<stdio.h>  #include<stdlib.h>  #include"mypgm\_s.h"  void posterization\_image()  /\* image1 を階調変換した画像を image2 へ \*/  {  int x, y, n, don, wk; /\* 制御変数 \*/  printf("画素値を任意の階級値の画像に変換します¥n");  printf("階級値 n=");  scanf\_s("%d", &n);  if (n < 2) printf("階級値を1以下にすることは出来ません。");  x\_size2 = x\_size1; /\* 横画素数 \*/  y\_size2 = y\_size1; /\* 縦画素数 \*/  for (y = 0; y < y\_size2; y++) {  for (x = 0; x < x\_size2; x++) {  don = image1[y][x] / (MAX\_BRIGHTNESS / n);  wk = (MAX\_BRIGHTNESS / (n - 1)) \* don;  if (wk> 255) wk = 255;  if (wk< 0) wk = 0;  image2[y][x] = (unsigned char)wk;  //image2[y][x] = image1[y][x] ;  }  }  }  void main(void)  {  load\_image\_data( ); /\* 画像データを image1 に読み込む \*/  posterization\_image( );  save\_image\_data( ); /\* image2 を保存する \*/  } |

ソース1　画像の明度を変更するC++プログラム

|  |  |
| --- | --- |
| また, このプログラムの実行結果を以下に示す。  図 11 変換前画像    図 12 posterization\_image関数処理後画像(n=2)    図 13 posterization\_image関数処理後画像(n=10) | 図 14変換前画像    図 15 posterization\_image関数処理後画像(n=2)    図 16 posterization\_image関数処理後画像(n=10) |

**課題２．アナログ画像をディジタル画像へ変換する方法を調べる。**

**考察**