画像処理実験 第4回

09430509
今田将也

2020年10月13日

1　cpuで「特徴点らしさ」画像を出力

資料の例を元に実装を完成させた．

DElem，Elemがimage.hに定義されていなかったので，第3回および第2回を元に移植させた．そして，Matrix構造体`Matrix *MatrixAlloc`についても移植をおこなった．

最後に，今回の要であるImageFeature関数についてだが，iyに関する計算．そして，iyy,ixxの$\sum$に相当する計算を行う処理を加えた．以下がその部分のソースコードである．

```
ix = IElem(im, x+u+1, y+v, 1) - IElem(im, x+u-1, y+v, 1);
iy = IElem(im, x+u, y+v+1, 1) - IElem(im, x+u, y+v-1, 1);
ixx += ix*ix; // ixx だけでなく ixy,iyy も計算する．
iyy += iy*iy;
ixy += ix*iy;
```

さらに，DElemには行列

$$\begin{pmatrix} i_{xx} & i_{xy} \\ i_{xy} & i_{yy} \end{pmatrix}$$

の小さい方の固有値を入れることで求める事ができるが，2行2列の行列の固有値は2次方程式の解の公式で求めることと同じであるため解の公式を使った．

```
DElem(im2,x,y) =
((ixx + iyy)+sqrt(pow(ixx+iyy,2.0)-4.0*(ixx*iyy-ixy*ixy)))/2.0;
```

Wで，自分の周りの上下左右の7マスを見て特徴点を取っている．この実装で出力された結果は図1に示した．このときのWは7である．処理時間は3.9Ghzのcpuで144msecだった．

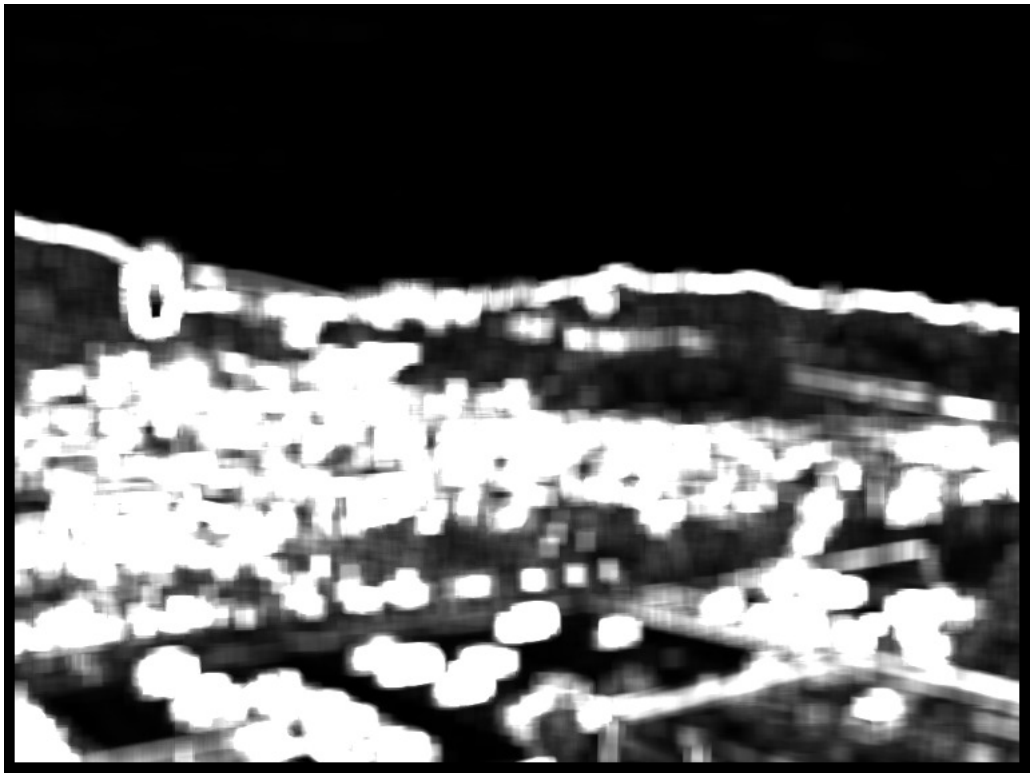

図 1: 出力結果

2 CPU での分解

資料を元に，3 重ループを 2 回行うことで実装した．資料には，特徴点検出の ixx,ixy,iyy は 15x15 の平滑化であり 225 個の和である．これを直接計算するより，横に隣接する 15 画素の和を中間配列に格納し，この配列の縦 15 要素の和によって最終結果を計算するとあるので，それを再現するようにした．出力結果は図 2 に示す．計算時間は 36msec であった．

```
int x,y,u,v,ix,iy;
for(y=1;y<im->H-1;y++) for(x=W+1;x<im->W-W-1;x++){
      double ixx,ixy,iyy;
      ixx=iyy=ixy=0;
      for(u=-W;u<=W;u++){
        ix = IElem(im, x+u+1, y, 1) - IElem(im, x+u-1, y, 1);
        iy = IElem(im, x+u, y+1, 1) - IElem(im, x+u, y-1, 1);
        ixx += ix*ix;
        iyy += iy*iy;
        ixy += ix*iy;
  }
```

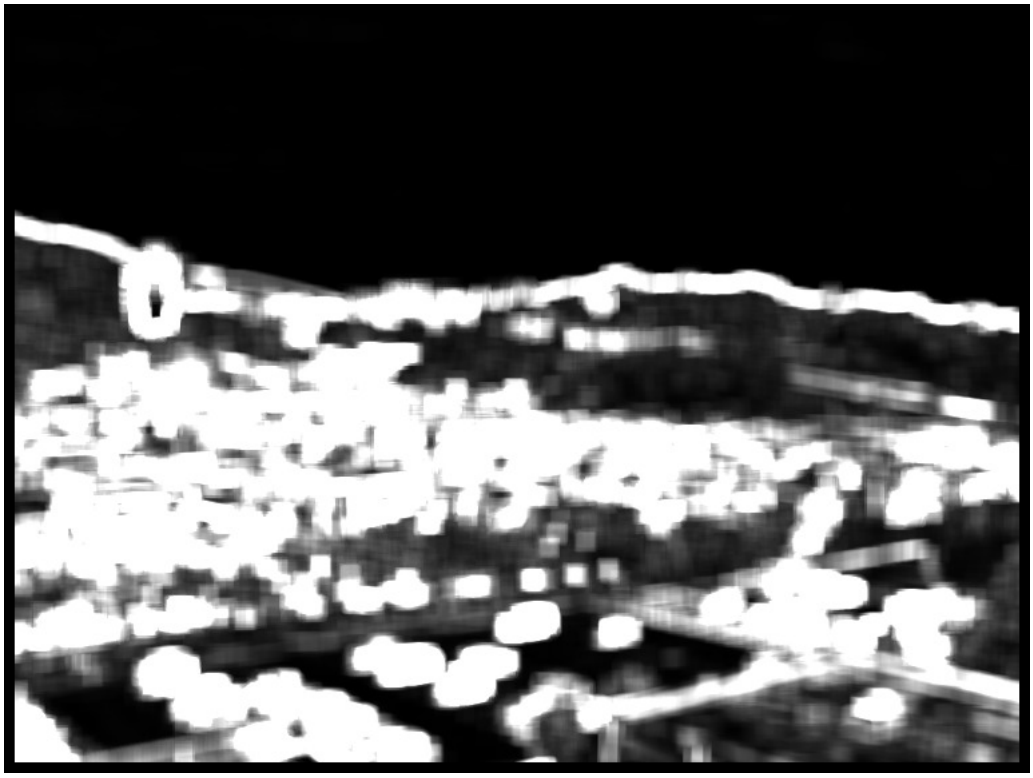

図 2: 出力結果

```
  DElem(im3,x,y) = ixx;
  DElem(im4,x,y) = ixy;
  DElem(im5,x,y) = iyy;
}
  for(y=W+1;y<im->H-W-1;y++) for(x=W+1;x<im->W-W-1;x++){
      double ixx,ixy,iyy;
      ixx=iyy=ixy=0;
      for(v=-W;v<=W;v++){
        ixx += DElem(im3, x, y+v);
        ixy += DElem(im4, x, y+v);
        iyy += DElem(im5, x, y+v);
  }
  DElem(im2,x,y) =
  ((ixx + iyy)+sqrt(pow(ixx+iyy,2.0)-4.0*(ixx*iyy-ixy*ixy)))/2.0;
}
```

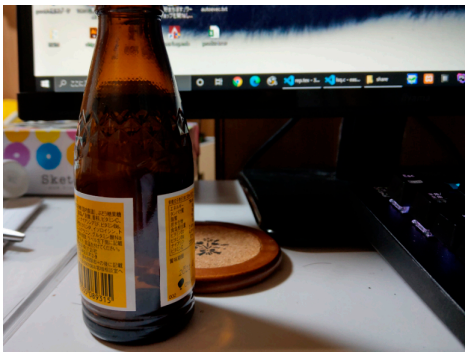

図 3: 元画像

図 4: 出力結果

3　実行時間の比較

CPU と CPU 分解の 2 つの時間を示す.

方法	時間 (msec)
CPU	144
CPU 分解	34

4　自分で撮影した画像で実験

作成した C プログラムに，図 3 を与えて出力された結果を図 4 に示した.

5　W の値をコマンドラインから指定

まず，image.h の ImageDrowBox の宣言に W を整数型で追加した．そして，使用する際にコマンドライン引数を atoi を使い，整数型に変えることで対応した.

```
ImageDrawBox(im,kk[i][0],kk[i][1],atoi(av[2]));
```

6　W の値を変えて複数回実行

w の値を 0 から 10 まで変えて画像の出力をした．数字が大きくなるにつれて処理時間も増えていった．図 5 から図 15 がその出力結果である．W=2 から W=4 あたりが綺麗に輪郭が出ている．W=7 以上は輪郭が太くなりすぎてあまり綺麗でなかった.

図 5: W=0

図 6: W=1

7　感想

CPU による実装を完了させることはでき，W の値を変えることでどのくらい変化があるのかということを画像を並べて比較することができた．GPU による実装はできてない上，分解についてもとりあえず実装し，動きまで理解ができていない．

図 7: W=2

図 8: W=3

図 9: W=4

図 10: W=5

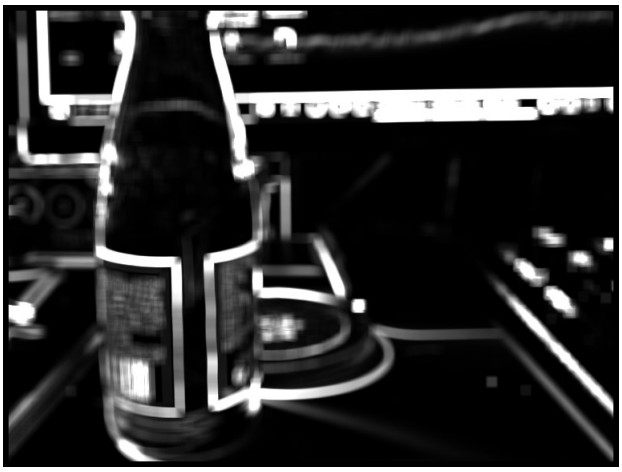

図 11: W=6

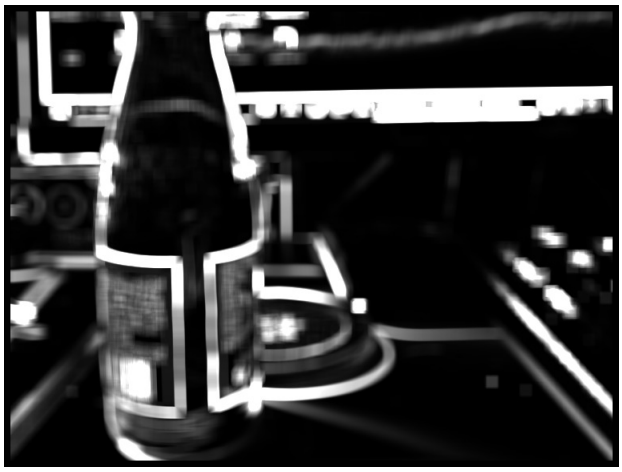

図 12: W=7

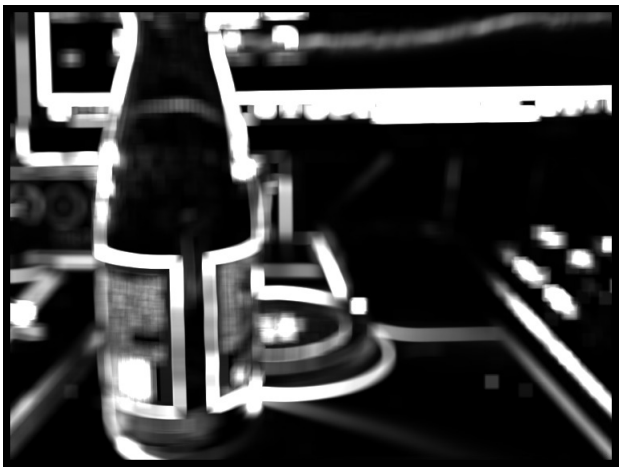

図 13: W=8

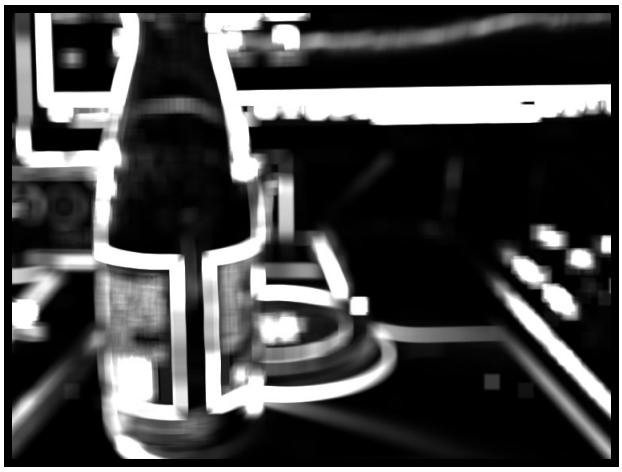

図 14: W=9

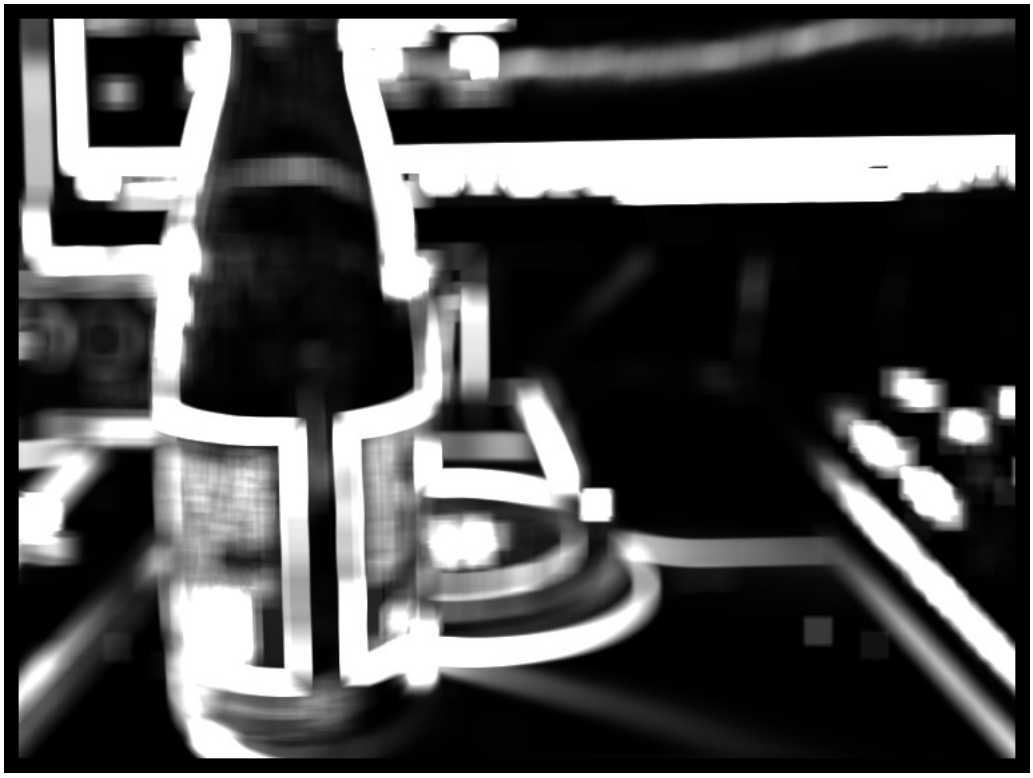

図 15: W=10