画像工学特論自由課題

学籍番号：093750

氏名：渡邉雄大

課題．特徴点を検出するプログラムを作成せよ．

1. プログラムについて

エッジ検出及び特徴点を検出するプログラムを作成した．エッジの検出は，画像の色情報から，注目するドットと隣接するドットの色の差を求め，しきい値を定めそれ以上の差があればエッジとして検出している．特徴点の抽出は，注目するドットの周囲のドットの色情報を取得し，周囲の色情報の値の平均と注目する座標の色情報の値の差を求め，特徴点の抽出を行っている．言語はHSPを用い，動作はWindows VistaとWindows 7上で確認した．

1. 実行結果と考察

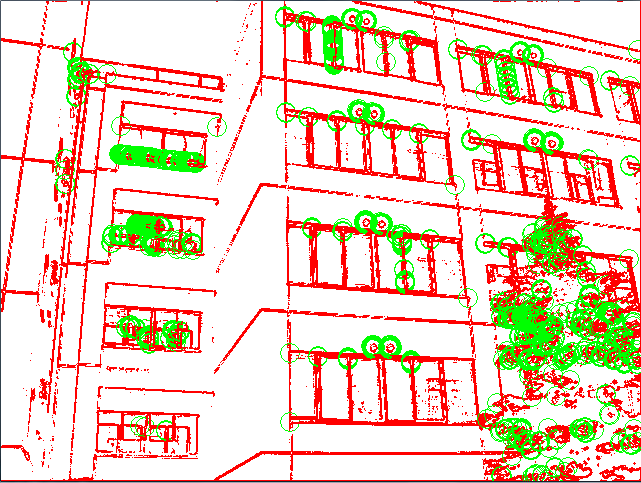


図１　サンプル画像aと実行結果

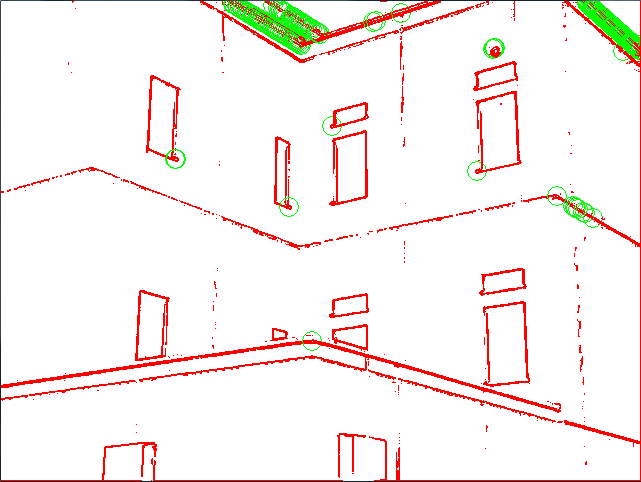


図２　サンプル画像bと実行結果



図２　サンプル画像cと実行結果

赤い点がエッジとして検出した点，緑の円が特徴点である．

エッジ検出は明度の違う面同士が交わるところの検出が出来ていない．これは検出処理がドットの色の差のみに注目しているため，明度の違いの処理がされていないためである．

特徴点の検出は，周囲の色の平均から注目するドットの色を引いているだけなので精度は良くない．もっと周囲の色情報を集めるか，エッジ検出の結果を利用する，色変化の勾配を利用すれば，精度が上がると考えられる．

1. ソースプログラム

|  |
| --- |
| dialog "bmp|ppm|pgm",16,"マップファイル"  if stat : fname = refstr :else: fname = "sample.bmp"  picload fname  g\_x = ginfo(12) //画像の横幅  g\_y = ginfo(13) //画像の縦幅  screen 1,200,40  color 0,0,0:font "systemfont",40,0:mes "処理中"  gsel 0  // 色情報格納用2次元配列  dim cor,1000,1000  dim cor\_r,1000,1000  dim cor\_g,1000,1000  dim cor\_b,1000,1000  // しきい値  shikii = 1000  // 画像の色情報の取得  for i,0,g\_x,1  for j,0,g\_y,1  pget i,j //色情報取得関数  cor\_r(i,j) = ginfo\_r //赤  cor\_g(i,j) = ginfo\_g //緑  cor\_b(i,j) = ginfo\_b //青  cor(i,j) = ginfo\_r + ginfo\_g + ginfo\_b  next  next  dim bb,g\_x,g\_y  dim sa\_check,8  // 周りのドットと比較する  for i,0,g\_x,1  for j,0,g\_y,1  pget i,j  if(i>0){ //左  sa\_1 = cor(i-1,j) - (ginfo\_r + ginfo\_g + ginfo\_b)  sa\_1 = sa\_1 \* sa\_1  sa\_r1 = cor\_r(i-1,j) - ginfo\_r  sa\_r1 = sa\_r1 \* sa\_r1  sa\_g1 = cor\_r(i-1,j) - ginfo\_g  sa\_g1 = sa\_g1 \* sa\_g1  sa\_b1 = cor\_b(i-1,j) - ginfo\_b  sa\_b1 = sa\_b1 \* sa\_b1  }  if(j>0){ //上  sa\_2 = cor(i,j-1) - (ginfo\_r + ginfo\_g + ginfo\_b)  sa\_2 = sa\_2 \* sa\_2  sa\_r2 = cor\_r(i,j-1) - ginfo\_r  sa\_r2 = sa\_r2 \* sa\_r2  sa\_g2 = cor\_r(i,j-1) - ginfo\_g  sa\_g2 = sa\_g2 \* sa\_g2  sa\_b2 = cor\_b(i,j-1) - ginfo\_b  sa\_b2 = sa\_b2 \* sa\_b2    }  if(i<g\_x){ //右  sa\_3 = cor(i+1,j) - (ginfo\_r + ginfo\_g + ginfo\_b)  sa\_3 = sa\_3 \* sa\_3  sa\_r3 = cor\_r(i+1,j) - ginfo\_r  sa\_r3 = sa\_r3 \* sa\_r3  sa\_g3 = cor\_r(i+1,j) - ginfo\_g  sa\_g3 = sa\_g3 \* sa\_g3  sa\_b3 = cor\_b(i+1,j) - ginfo\_b  sa\_b3 = sa\_b3 \* sa\_b3    }  if(j<g\_y){ //下  sa\_4 = cor(i,j+1) - (ginfo\_r + ginfo\_g + ginfo\_b)  sa\_4 = sa\_4 \* sa\_4  sa\_r4 = cor\_r(i,j+1) - ginfo\_r  sa\_r4 = sa\_r4 \* sa\_r4  sa\_g4 = cor\_r(i,j+1) - ginfo\_g  sa\_g4 = sa\_g4 \* sa\_g4  sa\_b4 = cor\_b(i,j+1) - ginfo\_b  sa\_b4 = sa\_b4 \* sa\_b4    }  //斜め方向  if(i>0 & j> 0){ //左上  sa\_5 = cor(i-1,j-1) - (ginfo\_r + ginfo\_g + ginfo\_b)  sa\_5 = sa\_1 \* sa\_1  sa\_r5 = cor\_r(i-1,j-1) - ginfo\_r  sa\_r5 = sa\_r5 \* sa\_r5  sa\_g5 = cor\_g(i-1,j-1) - ginfo\_g  sa\_g5 = sa\_g5 \* sa\_g5  sa\_b5 = cor\_b(i-1,j-1) - ginfo\_b  sa\_b5 = sa\_b5 \* sa\_b5  }  if(j>0 & i<g\_x){ //右上  sa\_6 = cor(i+1,j-1) - (ginfo\_r + ginfo\_g + ginfo\_b)  sa\_6 = sa\_6 \* sa\_6  sa\_r6 = cor\_r(i+1,j-1) - ginfo\_r  sa\_r6 = sa\_r6 \* sa\_r6  sa\_g6 = cor\_g(i+1,j-1) - ginfo\_g  sa\_g6 = sa\_g6 \* sa\_g6  sa\_b6 = cor\_b(i+1,j-1) - ginfo\_b  sa\_b6 = sa\_b6 \* sa\_b6  }  if(i<g\_x & j<g\_y){ //右下  sa\_7 = cor(i+1,j+1) - (ginfo\_r + ginfo\_g + ginfo\_b)  sa\_7 = sa\_7 \* sa\_7  sa\_r7 = cor\_r(i+1,j+1) - ginfo\_r  sa\_r7 = sa\_r7 \* sa\_r7  sa\_g7 = cor\_g(i+1,j+1) - ginfo\_g  sa\_g7 = sa\_g7 \* sa\_g7  sa\_b7 = cor\_b(i+1,j+1) - ginfo\_b  sa\_b7 = sa\_b7 \* sa\_b7  }  if(j<g\_y & i>0){ //左下  sa\_8 = cor(i-1,j+1) - (ginfo\_r + ginfo\_g + ginfo\_b)  sa\_8 = sa\_8 \* sa\_8  sa\_r8 = cor\_r(i-1,j+1) - ginfo\_r  sa\_r8 = sa\_r8 \* sa\_r8  sa\_g8 = cor\_g(i-1,j+1) - ginfo\_g  sa\_g8 = sa\_g8 \* sa\_g8  sa\_b8 = cor\_b(i-1,j+1) - ginfo\_b  sa\_b8 = sa\_b8 \* sa\_b8  }  // 外周マスの色を取得  if(i-2>0 & j>0 & j<g\_y){  gai\_1 = cor(i-3,j) + cor(i-3,j-1) + cor(i-3,j+1)  gai\_r1 = cor\_r(i-3,j) + cor\_r(i-3,j-1) + cor\_r(i-3,j+1)  gai\_g1 = cor\_g(i-3,j) + cor\_g(i-3,j-1) + cor\_g(i-3,j+1)  gai\_b1 = cor\_b(i-3,j) + cor\_b(i-3,j-1) + cor\_b(i-3,j+1)  }else{  gai\_1 = (ginfo\_r + ginfo\_g + ginfo\_b)\*3  gai\_r1 = ginfo\_r\*3  gai\_g1 = ginfo\_g\*3  gai\_b1 = ginfo\_b\*3  }  if(j-2>0 & i>0 & i<g\_x){  gai\_2 = cor(i,j-3) + cor(i-1,j-3) + cor(i+1,j-3)  gai\_r2 = cor\_r(i,j-3) + cor\_r(i-1,j-3) + cor\_r(i+1,j-3)  gai\_g2 = cor\_g(i,j-3) + cor\_g(i-1,j-3) + cor\_g(i+1,j-3)  gai\_b2 = cor\_b(i,j-3) + cor\_b(i-1,j-3) + cor\_b(i+1,j-3)  }else{  gai\_2 = (ginfo\_r + ginfo\_g + ginfo\_b)\*3  gai\_r2 = ginfo\_r\*3  gai\_g2 = ginfo\_g\*3  gai\_b2 = ginfo\_b\*3  }  if(g\_x>i+2 & j>0 & j<g\_y){  gai\_3 = cor(i+3,j) + cor(i+3,j-1) + cor(i+3,j+1)  gai\_r3 = cor\_r(i+3,j) + cor\_r(i+3,j-1) + cor\_r(i+3,j+1)  gai\_g3 = cor\_g(i+3,j) + cor\_g(i+3,j-1) + cor\_g(i+3,j+1)  gai\_b3 = cor\_b(i+3,j) + cor\_b(i+3,j-1) + cor\_b(i+3,j+1)  }else{  gai\_3 = (ginfo\_r + ginfo\_g + ginfo\_b)\*3  gai\_r3 = ginfo\_r\*3  gai\_g3 = ginfo\_g\*3  gai\_b3 = ginfo\_b\*3  }  if(g\_y>j+2 & i>0 & i<g\_x){  gai\_4 = cor(i,j+3) + cor(i-1,j+3) + cor(i+1,j+3)  gai\_r4 = cor\_r(i,j+3) + cor\_r(i-1,j+3) + cor\_r(i+1,j+3)  gai\_g4 = cor\_g(i,j+3) + cor\_g(i-1,j+3) + cor\_g(i+1,j+3)  gai\_b4 = cor\_b(i,j+3) + cor\_b(i-1,j+3) + cor\_b(i+1,j+3)  }else{  gai\_4 = (ginfo\_r + ginfo\_g + ginfo\_b)\*3  gai\_r4 = ginfo\_r\*3  gai\_g4 = ginfo\_g\*3  gai\_b4 = ginfo\_b\*3  }  gai\_av = (gai\_1 + gai\_2 + gai\_3 + gai\_4)/12  gai\_rav = (gai\_r1 + gai\_r2 + gai\_r3 + gai\_r4)/12  gai\_gav = (gai\_g1 + gai\_g2 + gai\_g3 + gai\_g4)/12  gai\_bav = (gai\_b1 + gai\_b2 + gai\_b3 + gai\_b4)/12  gai\_sa = abs(gai\_av - (ginfo\_r + ginfo\_g + ginfo\_b))  gai\_rsa = abs(gai\_rav - ginfo\_r)  gai\_gsa = abs(gai\_gav - ginfo\_g)  gai\_bsa = abs(gai\_bav - ginfo\_b)    // 隣接マスと比べて、色の差が大きかったらエッジかコーナー  if(sa\_1 + sa\_2 + sa\_3 + sa\_4 > 3\*shikii & (sa\_r1 + sa\_r2 + sa\_r3 + sa\_r4 > shikii | sa\_g1 + sa\_g2 + sa\_g3 + sa\_g4 > shikii | sa\_b1 + sa\_b2 + sa\_b3 + sa\_b4 > shikii)){  // 角かどうか判定(外周と中央の差が一定以上)  if( gai\_sa > 240 | (gai\_rsa >100 | gai\_gsa >100 | gai\_bsa >100) ){  bb(i,j)=2  }else{  bb(i,j)=1  }  }  next  next  // エッジと特徴点を描画  screen 1,g\_x,g\_y  for i,0,g\_x,1  for j,0,g\_y,1  if(bb(i,j)=1):color 255,0,0:pset i,j  if(bb(i,j)=2):color 0,255,0:circle i-10,j-10,i+10,j+10,0  next  next |