# 画像応用数学特論レポート提出（1/14）

情報科学部　知能科学専攻
[redacted] 福本翔平 [redacted]@wm.hiroshima-cu.ac.jp

【課題】

・対応する 2 枚のステレオペア画像ファイルを読み込み，α 拡張アルゴリズムを使ってステレオマッチングを行い，その視差を画像ファイルにして出力する．

・対応する 2 枚のステレオペア画像ファイルを読み込み，階層グラフカットアルゴリズムを使ってステレオマッチングを行い，その視差を画像ファイルにして出力する．

以上の二つのプログラムを作成し，それらの精度を比較・考察する．

【PC 環境】

バージョン；ubuntu 13.04

メモリ：31.4GiB

プロセッサ：Intel®Xeon(R) CPU E3-1270 V2 @ 3.50GHz*8

【理論】

『α 拡張』

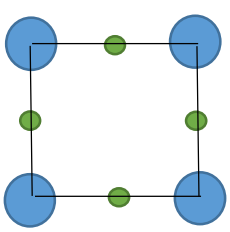
・$(x_l, y_l)$と$(x_r, y_r)$を左の画像と右の画像の明るさを比較して，最もぴったり合う視差を探す．差の計算にはいろいろな種類があり，今回は SAD を用いてコスト関数を計算する．またとりうる視差の最大値は 255 とする．

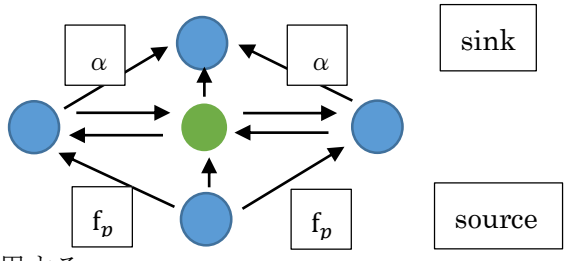
$$\sum_w \|I_l(x,y) - I_r(x-d,y)\|$$

またスムーズコストの値に関しては以下の式を適用する．

$$V(f_p, f_q) = c|f_p - f_q|$$

・ノードとエッジに関しては，読み込んだピクセルの間にノードとエッジを追加する．形式的にはピクセルの間に余分なノードを付け、それらとの間にエッジを付ける．






追加したノードにα拡張アルゴリズムを適用する．

『階層グラフカット』

・順々にαをループで回すα拡張と異なり，階層グラフカットではあるループにおいてラベル数$A[i]$のうちβ（現在のラベル）に近いものを選ぶ

$$A[0]=0, A[1]=\frac{n}{2}, A[2j]=\bigcup_{k=0}^{2^{j-1}-1}\left\{\frac{1+4k}{2^{j+1}}n\right\},\ A[2j+1]=\bigcup_{k=0}^{2^{j-1}-1}\left\{\frac{3+4k}{2^{j+1}}n\right\},$$

ラベル数における$A[i]$の個数は

$$\lceil A_i \rceil = max\left(1, 2^{\left[\frac{i}{2}\right]-1}\right)$$

【アルゴリズム】

『α拡張』

```
for ループ=0～とても大きな値
  success=0
  for a=0~DISPMAX
    グラフの初期化
    全てのノードの追加
    for p=全てのピクセル
      ノードをソース側に D(f_p)かシンク側 D(a)かを設定
    for (p,q)=全ての隣接点
      ノードaをソース側にV(f_p,f_q),シンク側に V(a,a)を設定
      ノードpとノードaのエッジの重みにV(f_p,a)を設定
      ノードaとノードpのエッジの重みにV(a,f_q)を設定
    最大流・最小カットアルゴリズムの適用
    E'=求めたラベルで計算した総ソスト関数
    E'<Eなら現在のラベルを求めたラベルに変更し，success=1にする
    グラフ消去
  もしsuccess==0ならループを脱出
```

『階層グラフカット』

```
for ループ=0～とても大きな値
  success=0
  for a=0~11
    グラフの初期化
    全てのノードの追加
    for p=全てのピクセル
      A[i]のうち，b_pに最も近い値をa_pに設定
```

ノードをソース側に $D(b_p)$かシンク側 $D(a_p)$かを設定

for (p,q)=全ての隣接点

$A[i]$のうち，$b_p$に最も近い値を$a_p$に設定

$A[i]$のうち，$b_q$に最も近い値を$a_q$に設定

ノード a をソース側に$V(b_p,b_q)$かシンク側$V(a_p,a_q)$かを設定

もし$V(b_p,b_q) \leq V(a_p,a_q)$なら

ノード a からノード p へのエッジの重みに 10000 を設定

ノード a からノード q へのエッジの重みに 10000 を設定

ノード p からノード a へのエッジの重みに$V(a_p,b_q) - V(b_p,b_q)$又は 0 を設定

ノード q からノード a へのエッジの重みに$V(b_p,a_q) - V(b_p,b_q)$又は 0 を設定

もし$V(b_p,b_q) \geq V(a_p,a_q)$なら

ノード p からノード q へのエッジの重みに 10000 を設定

ノード q からノード a へのエッジの重みに 10000 を設定

ノード a からノード q へのエッジの重みに$V(a_p,b_q) - V(a_p,a_q)$又は 0 を設定

ノード a からノード p へのエッジの重みに$V(b_p,a_q) - V(a_p,a_q)$又は 0 を設定

最大流・最小カットアルゴリズムの適用

E'=求めたラベルで計算した総ソスト関数

E'<E なら現在のラベルを求めたラベルに変更し，success=1 にする

グラフ消去

もし success==0 ならループを脱出

【過程】

・プログラム動作には二つの画像 view0.png と view06.png（両者とも png ファイル）が必要．また画像に関する関数を用いる Customing.h を利用し，プログラム内で include を行っている．

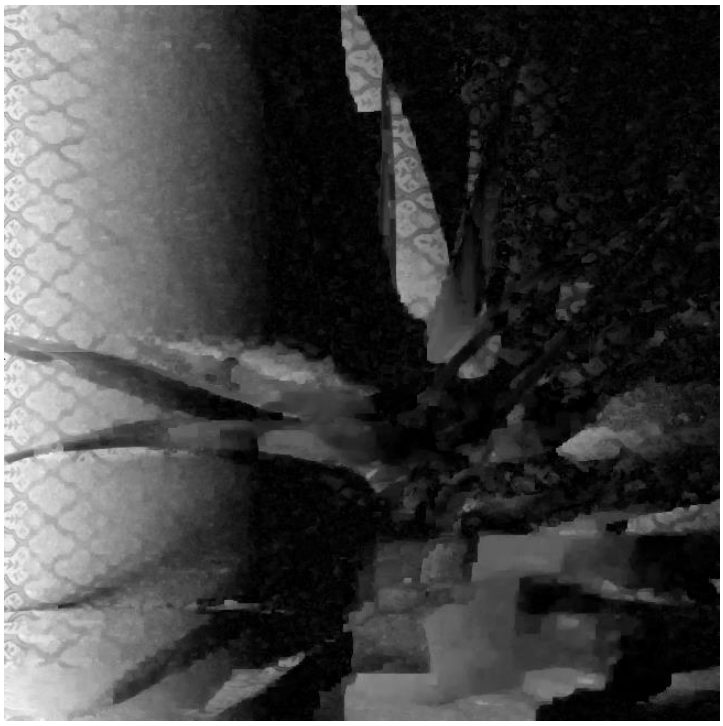
・コンパイル方法は以下の通り

g++ -o compile4 compile4.cpp graph.cpp maxflow.cpp CustomImg.cpp `pkg-config --cflags opencv` `pkg-config --libs opencv`

・実行は./1 或いは./compile4 で行う

【結果】

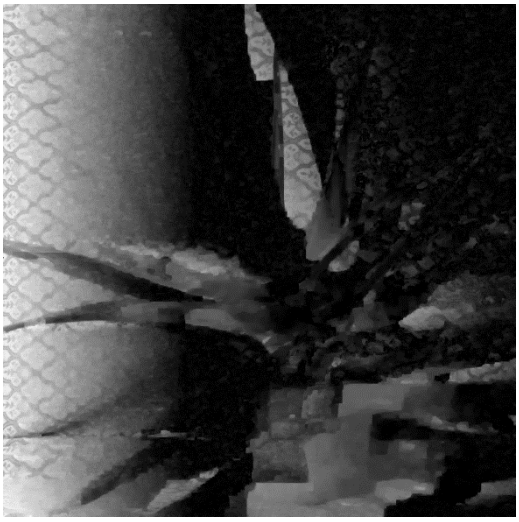
『650*650 ピクセルの画像の場合』



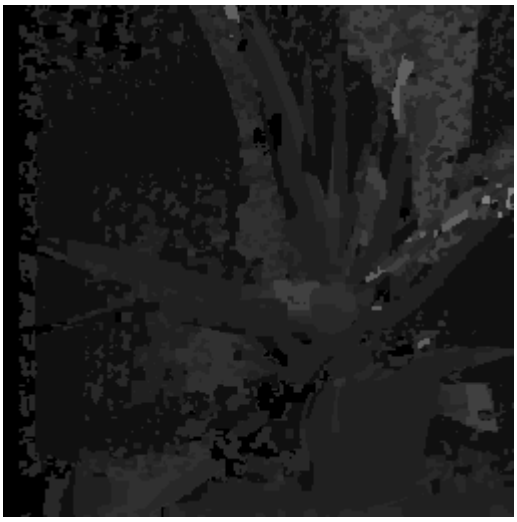
α 拡張の視差画像（1075.21ms）

階層グラフカットの視差画像（16.01ms）

『250*250 ピクセルの画像の場合』



α 拡張の視差画像（129.35sec）



階層グラフカットの視差画像（1.78ms）

【考察】

α 拡張による視差画像はどちらも背景が上手く表示されずに留まった．また元画像である画像もそこまで良くはない．階層グラフカットに関しては画像のピクセル値が大きいと比較的上手く出力されるが，小さい画像を実行すると全体的に暗めの出力がされてしまう．恐らくプログラムの誤差だと思われる．また実行時間からすると階層グラフカットの方が短いが，精度はどちらもそこまで良くはない．ノードなどの接続が上手く出来ていない可能性はあるが，どちらにせよプログラムの改良が必要だと思われる．