画像応用数学特論　第 6 回レポート課題

出題日 12 月 15 日 (木)
締切日 1 月 23 日 (火)
提出日 1 月 23 日 (火)

[redacted] 石丸功一
ishimaru@cv.info.hiroshima-cu.ac.jp

課題　階層グラフカットでステレオマッチング

本課題では，階層グラフカットでステレオマッチングを行う．入力画像は図 1，図 2 である．この入力画像は画像サイズを 256 × 222[pixel] にリサイズし、グレースケール化してある．

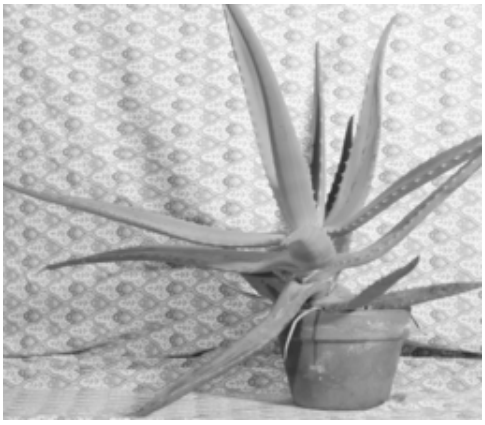


図 1: 左側から撮影した入力画像

図 2: 右側から撮影した入力画像

実験環境

- OS : UbuntuStudio11.10
- CPU : intel Core2 Duo E8500
- メモリ : 2GB
- コンパイラ : g++ (Ubuntu/Linaro 4.4.4-14ubuntu5) 4.4.5
- 使用言語 : C 言語
- ライブラリ : MAXFLOW，OpenCV

実行方法

添付ファイルを解凍して，ディレクトリ a_graphcut に入っているプログラムが α 拡張グラフカットを行うプログラムです．また，ディレクトリ h_graphcut に入っているプログラムが階層グラフカットを行うプログラムです．どちらのディレクトリにもディレクトリ img があり，図 1，2 の 2 枚を入れてあります．

コンパイル法方は，Makefile があるので make コマンドでコンパイルできます.

実行法方は，./graphcut 左側から撮影した画像 右側から撮影した画像 最大視差の値 定数 c_0 定数 c_1 閾値 T 出力画像名 で実行できます．(例)./graphcut img/viewL.png img/viewR.png 32 100 30 100 img/output.png

アルゴリズム

1. 現在の全ての画素のラベルを β_k で初期化
2. 階層ラベル A を作成
3. グラフの総コスト E を非常に大きい値に初期化
4. 総コスト E が収束するまで以下をループ
   (a) 階層ラベルの深さ *depth* だけ以下をループ
      i. グラフを初期化
      ii. ノードを追加
      iii. 全ての画素 k に対して
         A. 現在の深さ i の階層ラベル A[i] のうち最も β_k に最も近い値を α_k に設定
         B. ノードのソース側に D(β_k)，シンク側に D(α_k) を設定
      iv. 全ての隣接する画素 p, q に対して，
         A. 現在の深さ i の階層ラベル A[i] のうち最も $\beta[p]$ に最も近い値を α_p に設定
         B. 現在の深さ i の階層ラベル A[i] のうち最も $\beta[q]$ に最も近い値を α_q に設定
         C. もし，V(β_p, β_q) = V(α_p, α_q) ならば，
            ノード p からノード q へのエッジの重みに V(α_p, β_q) を設定
            ノード q からノード p へのエッジの重みに V(β_p, α_q) を設定
         D. もし，V(β_p, β_q) < V(α_p, α_q) ならば，
            画素 p, q 間のノード a のソース側に V(β_p, β_q)，シンク側に V(α_p, α_q) を設定
            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(α_p, β_q)−V(β_p, β_q) を設定
            ノード q からノード a へのエッジの重みに V(β_p, α_q)−V(β_p, β_q) を設定
         E. もし，V(β_p, β_q) > V(α_p, α_q) ならば
            画素 p, q 間のノード a のソース側に V(β_p, β_q)，シンク側に V(α_p, α_q) を設定
            ノード p からノード a 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 p へのエッジの重みに V(α_p, β_q)−V(α_p, α_q) を設定
            ノード a からノード q へのエッジの重みに V(β_p, α_q)−V(α_p, α_q) を設定
      v. 最大流・最小カットアルゴリズムを適用し，総コスト E' を計算
      vi. もし，E ¿ E’ ならば，
         A. 現在のラベル β_k を求まったラベル α_k に更新する.
         B. 総コストの更新 ($E = E'$)

vii. グラフの消去

実行結果１　スムーズコストに定数 c で行った場合

データコスト $D(f_{x,y})$ とスムーズコスト $V(a,b)$ を次式で定義する．ただし，c は定数である．

$$\begin{aligned} D(f_{x,y}) &= \Sigma w_{x,y}||l_l(x,y) - l_r(x - f_{x,y}, y)|| \\ V(a,b) &= c|f_p - f_q| \end{aligned} \tag{1}$$

ここで，最適な定数 c を求めるため，定数 c を 10 から 60 まで 10 ずつ増加させ，ステレオマッチングを行った．実行結果を図 3 に載せる．

考察１

図 3(a)，(b) のように定数 c が小さいと視差が滑らかに変化していない箇所がある (画像の左側部分)．逆に図 3(e)，(f) のように大きいと視差の変化が細かく取れていない．今回の入力データでは，定数 $c = 40$ が最も良い結果が得られている．

実行結果２　スムーズコストにエッジの強さを考慮した場合

データコストとスムーズコストを次式で定義する．ただし，$c0, c1, T$ は定数である．

$$\begin{aligned} D(f_{x,y}) &= \Sigma w_{x,y}||l_l(x,y) - l_r(x - f_{x,y}, y)|| \\ V(a,b) &= c|f_p - f_q| \\ c &= \begin{cases} c0 & if \quad d > T \\ c1 & if \quad d \geq T \end{cases} \end{aligned}$$

d は画素 (x,y) のエッジの強さであり，次のフィルタをかけて求める．

-2	-6	-2
-6	32	-6
-2	-6	-2

最適な定数の値を調べるために定数の値を少しずつ変化させてステレオマッチングを行った．最も良い結果が得られたのが，$c_0 = 100, c_1 = 30, T = 100$ であり，結果を図 4 に示す．

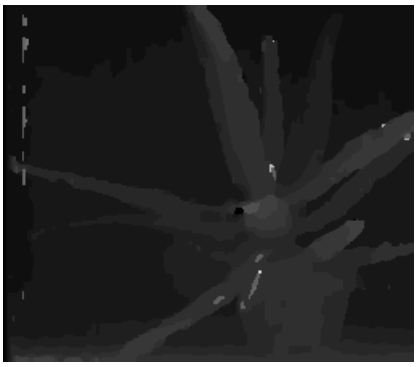
考察２

実行結果より，エッジの強さを考慮しなかった場合の図 3(d) と同じ用な結果が得られている．

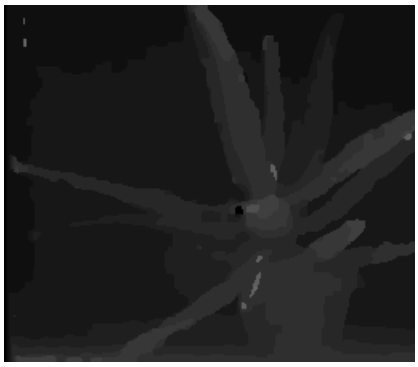
実行結果３　計算時間の比較

最後に計算時間の比較を行う．α 拡張グラフカットでスムーズコストにエッジの強さを考慮した場合と考慮しなかった場合の 2 パターンと階層グラフカットでスムーズコストにエッジの強さを考慮した場合と考慮しなかった場合の 2 パターンの計 4 パターンの計算時間の比較を行った．

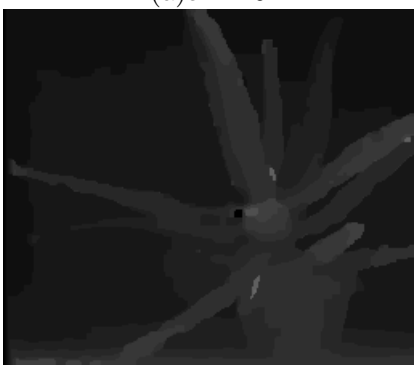
計測の方法は同じ入力データとパラメータでステレオマッチングを 10 回行い，その平均値を計算時間とした．結果を以下に載せる．



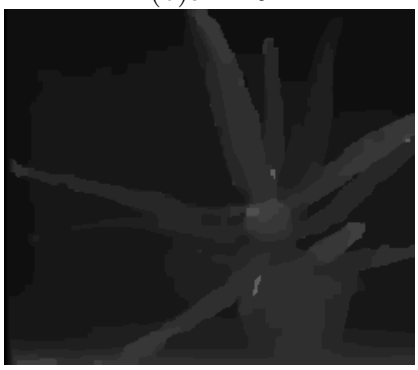
(a)$c = 10$



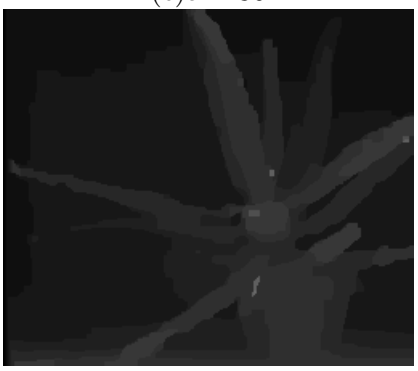
(b)$c = 20$



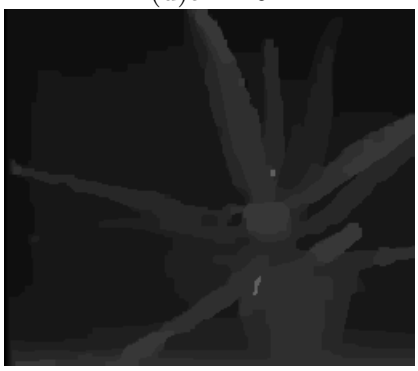
(c)$c = 30$



(d)$c = 40$



(e)$c = 50$



(f)$c = 60$

図 3: 定数 c の場合の実行結果

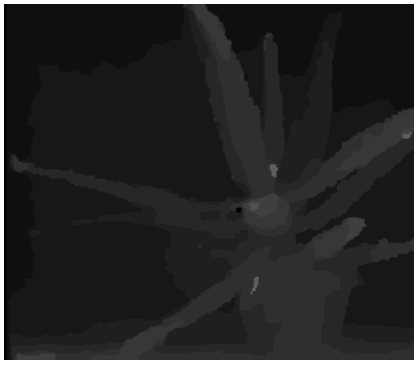


図 4: エッジの強さを考慮した場合の実行結果 ($c_0 = 100, c_1 = 30, T = 100$)

グラフカットの種類	コストの計算式	実行時間 [sec]
α 拡張	エッジの強さを考慮しない	46.75
α 拡張	エッジの強さを考慮する	45.63
階層	エッジの強さを考慮しない	12.13
階層	エッジの強さを考慮する	10.92

考察３

階層グラフカットの方が α 拡張グラフカットよりも 4 倍も高速にステレオマッチングできることが分かる．また，少しではあるがエッジの強さを考慮する方が高速であることが分かった．