

画像応用数学特論
第6回レポート
階層グラフカットでステレオ


知能工学専攻

坪石 健志

tsuboishi@ime.info.hiroshima-cu.ac.jp

■環境

- ・ OS : OSX 10.9.5
- ・ プロセッサ : 2.3GHz Intel Core i7
- ・ メモリ : 8GB
- ・ 開発環境 : Xcode 6.0.1
- ・ ライブラリ : OpenCV , MAXFLO
- ・ プログラミング言語 : C/C++

■アルゴリズム

階層グラフカットでステレオマッチングを行ったアルゴリズムの大きな流れを以下に示す。

- ①ラベルの生成
- ②右からと左からの2枚の画像を読み込む
- ③グラフの初期化
- ④全画素に対してノードの追加、 A_p を決定、ソースとシンクのデータコストを設定を行う。
- ⑤隣接する画素に対してノードの追加、 A_p, A_q を決定、ソースとシンクと隣接画素のスムーズコストを設定を行う。
- ⑥最大流最小カットを行う。
- ⑦総コストが2周目以降前回より小さければ③を繰り返す
- ⑧画像の出力

アルファ拡張でステレオマッチングを行ったアルゴリズムの大きな流れを以下に示す

- ①右からと左からの2枚の画像を読み込む
- ②グラフの初期化
- ③全画素に対してノードの追加、ソースとシンクのデータコストを設定を行う。
- ④隣接する画素に対してノードの追加、ソースとシンクと隣接画素のスムーズコストを設定を行う。
- ⑤最大流最小カットを行う。
- ⑥総コストが2周目以降前回より小さければ③を繰り返す
- ⑦画像の出力

階層グラフカットのコストを図1、アルファ拡張のコストを図2に示す

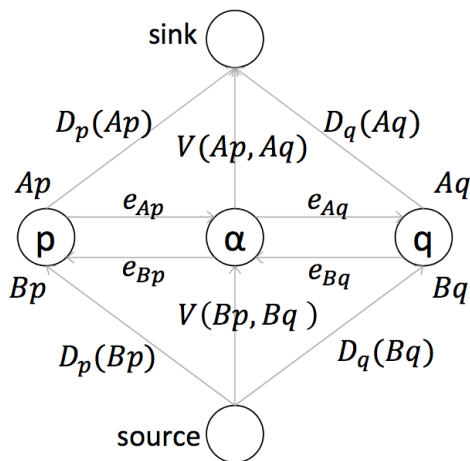


図1：階層グラフカットのコスト

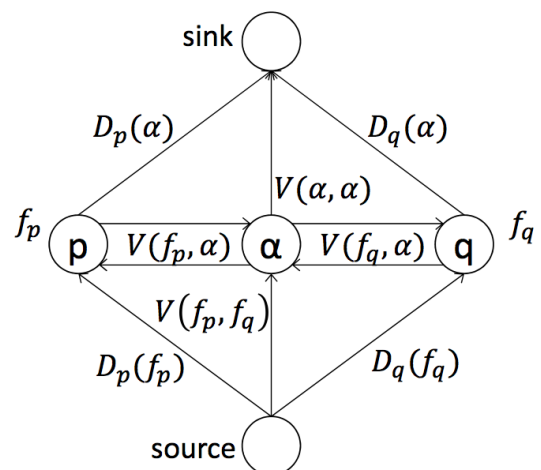


図2：アルファ拡張のコスト

■実験

階層グラフカットとアルファ拡張で、それぞれのウィンドウサイズでステレオマッチングを行った結果を以下に示す。

データコストの計算：SAD

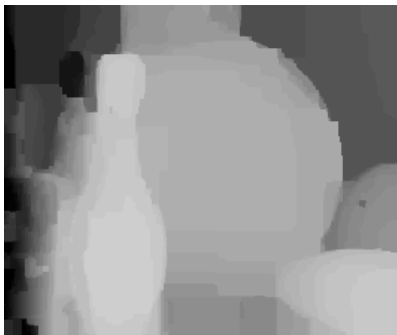


右からの画像



左からの画像

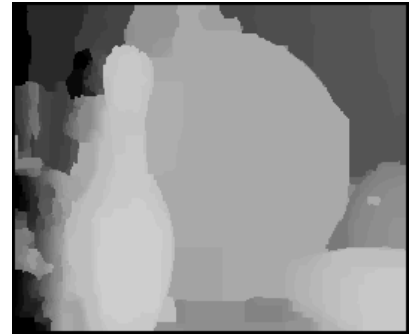
●階層グラフカット



ウィンドウサイズ 1x1



ウィンドウサイズ 3x3

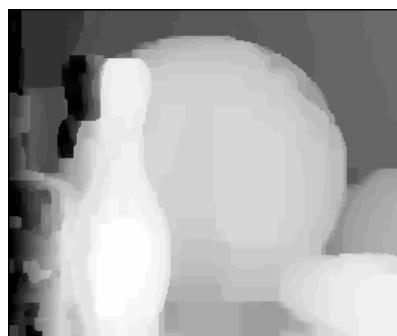


ウィンドウサイズ 5x5

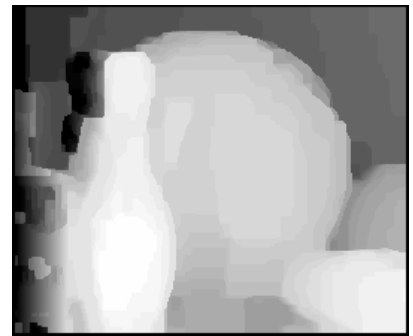
●アルファ拡張



ウィンドウサイズ 1x1



ウィンドウサイズ 3x3



ウィンドウサイズ 5x5

●実行速度

	アルファ拡張	階層グラフカット
ウィンドウ 1x1	104.614143 [s]	14.198378 [s]
ウィンドウ 3x3	59.153503 [s]	9.048819 [s]
ウィンドウ 5x5	67.947196 [s]	13.312526 [s]

■考察

階層グラフカットとアルファ拡張のステレオマッチングの結果を比較すると、やはりアルファ拡張の方が球を上手く取れていたり手前から奥に向けて濃度の変化が見られるので、精度が良いと思われる。ウィンドウサイズについては、階層グラフカットでは大きくなるにつれて濃度の変化が少なくなっていた。アルファ拡張では濃度の変化は細かくなっているが、球と背景の境界が正しく表示されていない。よって、ウィンドウサイズはなるべく小さい方が精度が良いと考えられる。実行速度については、アルファ拡張に比べ階層グラフカットは非常に早く処理を行えている。また、ウィンドウサイズで比較すると 1x1、5x5、3x3 の順に処理時間がかかっていた。

以上より、階層グラフカットでは高速で処理することに向いており、アルファ拡張は速度こそ遅いものの精度が良い。作成したプログラムではウィンドウサイズ 3x3 が最も早く、ウィンドウサイズ 1x1 が最も精度が良いという事が分かった。