1N-07

キャッシュ効率を考慮したガイデットフィルタの高速化

中村　将大 †　　前田 慶博 †　　福嶋 慶繁 †
† 名古屋工業大学

1　はじめに

エッジ保持平滑化フィルタの一つであるガイデットフィルタ [1] はフィルタカーネル内の局所線形性を用いることで画像の平滑化処理を行うフィルタであり，画像のボケやグラデーションなどの勾配のある画素に対して効果的なフィルタである．このフィルタは，複数の画像において注目画素を中心とする局所領域内の平均と分散を使用する．画像の平均と分散はボックスフィルタによる平滑化処理によって求めることができるため，従来手法によるガイデットフィルタ実装ではボックスフィルタが複数回使用されている．

本稿では，キャッシュ効率を考慮して，並列化処理のパターンを最小単位としてまとめ，ベクトル化することで効率的なガイデットフィルタの実装を行う．具体的には，ボックスフィルタを画像毎に用いるのではなく，処理する画像を同一ループ内にまとめ，同時に処理することでループ回数を削減する．さらに，ボックスフィルタと別の演算との結合をすることで，演算強度を高める．

2　ガイデットフィルタ

ガイデットフィルタの出力画像は，ガイド画像の適当な線形変換であるという仮定のもとで与えられる．ガイド画像 I がグレースケール画像である場合，出力画像 q は以下の式で求められる．

$$q_{\boldsymbol{i}} = \bar{a}_{\boldsymbol{i}} I_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}} \tag{1}$$

$$\bar{a}_{\boldsymbol{i}} = \frac{1}{|\omega|}\sum_{\boldsymbol{k}\in\omega_{\boldsymbol{i}}} a_{\boldsymbol{k}}, \quad \bar{b}_{\boldsymbol{i}} = \frac{1}{|\omega|}\sum_{\boldsymbol{k}\in\omega_{\boldsymbol{i}}} b_{\boldsymbol{k}} \tag{2}$$

ここで，$\omega_{\boldsymbol{i}}$ は画素 $\boldsymbol{i}$ を中心とする周辺画素の集合，$|\omega|$ は $\omega_{\boldsymbol{i}}$ の総画素数，$a_{\boldsymbol{k}}$，$b_{\boldsymbol{k}}$ は式 (3)，(4) で求められる線形係数である．

$$a_{\boldsymbol{k}} = \frac{\frac{1}{|\omega|}\sum_{\boldsymbol{i}\in\omega_{\boldsymbol{k}}} I_{\boldsymbol{i}} p_{\boldsymbol{i}} - \mu_{\boldsymbol{k}}\bar{p}_{\boldsymbol{k}}}{\sigma_{\boldsymbol{k}}^2 + \epsilon} \tag{3}$$

$$b_{\boldsymbol{k}} = \bar{p}_{\boldsymbol{k}} - a_{\boldsymbol{k}}\mu_{\boldsymbol{k}} \tag{4}$$

"Acceleration of Guided Filtering for Cache Efficiency"
Masahiro Nakamura†, Yoshihiro Maeda†, Norishige Fukushima†
† Nagoya Institute of Technology

Algorithm 1 Conventional implementation

1: $\mathrm{mean}_I = f_{mean}(I, r)$
　$\mathrm{mean}_p = f_{mean}(p, r)$
　$\mathrm{corr}_I = f_{mean}(I. * I, r)$
　$\mathrm{corr}_{Ip} = f_{mean}(I. * p, r)$
2: $\mathrm{var}_I = \mathrm{corr}_I - \mathrm{mean}_I. * \mathrm{mean}_I$
　$\mathrm{cov}_{Ip} = \mathrm{corr}_{Ip} - \mathrm{mean}_I. * \mathrm{mean}_p$
3: $a = \mathrm{cov}_{Ip}./(\mathrm{var}_I + \epsilon)$
　$b = \mathrm{mean}_p - a. * \mathrm{mean}_I$
4: $\mathrm{mean}_a = f_{mean}(a, r)$
　$\mathrm{mean}_b = f_{mean}(b, r)$
5: $q = \mathrm{mean}_a. * I + \mathrm{mean}_b$

ここで，$\mu_{\boldsymbol{k}}$，$\sigma_{\boldsymbol{k}}^2$ は $\omega_{\boldsymbol{k}}$ におけるガイド画像 I の平均と分散，$\bar{p}_{\boldsymbol{k}}$ は $\omega_{\boldsymbol{k}}$ における入力画像 p の平均，ϵ は正規化パラメータである．また，ガイド画像 $\boldsymbol{I}$ がカラー画像の場合，出力画像 q は以下の式で求められる．

$$q_{\boldsymbol{i}} = \bar{\boldsymbol{a}}_{\boldsymbol{i}}^T \boldsymbol{I}_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}} \tag{5}$$

$$\bar{\boldsymbol{a}}_{\boldsymbol{i}} = \frac{1}{|\omega|}\sum_{\boldsymbol{k}\in\omega_{\boldsymbol{i}}} \boldsymbol{a}_{\boldsymbol{k}}, \quad \bar{b}_{\boldsymbol{i}} = \frac{1}{|\omega|}\sum_{\boldsymbol{k}\in\omega_{\boldsymbol{i}}} b_{\boldsymbol{k}} \tag{6}$$

ここで，$\boldsymbol{a}_{\boldsymbol{k}}$，$b_{\boldsymbol{k}}$ は式 (7)，(8) で求められる線形係数である．

$$\boldsymbol{a}_{\boldsymbol{k}} = (\boldsymbol{\Sigma}_{\boldsymbol{k}} + \epsilon\boldsymbol{U})^{-1}\left(\frac{1}{|\omega|}\sum_{\boldsymbol{i}\in\omega_{\boldsymbol{k}}} \boldsymbol{I}_{\boldsymbol{i}} p_{\boldsymbol{i}} - \boldsymbol{\mu}_{\boldsymbol{k}}\bar{p}_{\boldsymbol{k}}\right) \tag{7}$$

$$b_{\boldsymbol{k}} = \bar{p}_{\boldsymbol{k}} - \boldsymbol{a}_{\boldsymbol{k}}^T \boldsymbol{\mu}_{\boldsymbol{k}} \tag{8}$$

ここで，$\boldsymbol{\Sigma}_{\boldsymbol{k}}$ は $\omega_{\boldsymbol{k}}$ におけるガイド画像 $\boldsymbol{I}$ の 3 行 3 列の共分散行列，$\boldsymbol{U}$ は 3 行 3 列の単位行列である．

ガイデットフィルタの疑似コードを Algorithm 1 に示す．ここで，$f_{mean}(\cdot, r)$ はカーネル半径 r のボックスフィルタによる平滑化処理であり，$.*$ は配列のアダマール積を表す演算子，$./$ は配列の右除算を表す演算子である．また，入力画像 p がカラー画像のとき，チャネル毎に分けて処理を 3 回繰り返す．このように，ガイデットフィルタの実装は，ボックスフィルタを複数の画像に対して行うことによって，大半の処理が行われる．

3　提案手法

提案手法では，ボックスフィルタ処理を画像毎に行うのではなく，同一ループ内で同時に処理する

ことでループ回数の削減と演算の結合を行う．提案手法によるガイデットフィルタ実装の疑似コードを Algorithm 2 に示す．なお，今回の実装において，ボックスフィルタはフィルタカーネルの可分性を用いてサムドエリアテーブル [2] による実装を行う．Algorithm 1 における $f_{mean}(\cdot, r)$ は，カーネルサイズ $1 \times (2r+1)$ の水平方向の総和を求める $f_{RowSum}(\cdot, r)$ と，カーネルサイズ $(2r+1) \times 1$ の垂直方向の総和を求める $f_{ColumnSum}(\cdot, r)$ に分割することができる．

画像同士の四則演算はマップパターンであり，ボックスフィルタの可分性を使ったサムドエリアテーブルはレデュースパターンである [3]．マップの後にレデュースが続き依存関係がない場合，マップとレデュースは融合することで最適化される．これにより並列性を維持したまま，キャッシュ効率の高い実装が可能となる．

具体的に提案手法では，ある画素における p, I, I^2, Ip のそれぞれの平均値を同時に求める．また，同一ループ内で複数枚の画像に処理をするため，Algorithm 1 では次のステップで行われる計算に必要な情報がすべてそろう．よって，提案手法では $\mathrm{var}_I, \mathrm{cov}_{Ip}$ を求める演算との結合が可能となり，$\mathrm{mean}_I, \mathrm{mean}_p, \mathrm{corr}_I, \mathrm{corr}_{Ip}$ を求める演算とほぼ同時に求めることができる．同様に Algorithm 1 のステップ 3 の a, b とステップ 5 の q も $f_{ColumnSum}(\cdot, r)$ とほぼ同時に求めることが可能となる．これによってループ回数が削減されるとともに，データの局所性が高まり，キャッシュ効率の高い実装が可能となる．

4 実験

実験では，Algorithm 1 と提案手法である Algorithm 2 によるガイデットフィルタ実装との速度比較を行った．図 1 に処理時間の結果を示す．Algorithm 1 による実装よりも，提案手法による実装が高速に動作していることが分かる．

実験に使用した入力，ガイド画像の解像度は，2268×1512 でグレースケールとカラー画像の組み合わせであり，フィルタに用いたカーネル半径は 10 である．また今回はマルチコア並列化を行わず，ベクトル化のみで実験を行った．

5 おわりに

本稿では，キャッシュ効率を考慮したガイデットフィルタの実装として，ボックスフィルタを同一ループ内で同時に処理することで，ループ回数の削減と

Algorithm 2 Proposed implementation

1: $\mathrm{mean}_{I_H} = f_{RowSum}(I, r)$
$\mathrm{mean}_{p_H} = f_{RowSum}(p, r)$
$\mathrm{corr}_{I_H} = f_{RowSum}(I. * I, r)$
$\mathrm{corr}_{Ip_H} = f_{RowSum}(I. * p, r)$
2: $\mathrm{mean}_I = f_{ColumnSum}(\mathrm{mean}_{I_H}, r)$
$\mathrm{mean}_p = f_{ColumnSum}(\mathrm{mean}_{p_H}, r)$
$\mathrm{corr}_I = f_{ColumnSum}(\mathrm{corr}_{I_H}, r)$
$\mathrm{corr}_{Ip} = f_{ColumnSum}(\mathrm{corr}_{Ip_H}, r)$
$\mathrm{var}_I = \mathrm{corr}_I - \mathrm{mean}_I. * \mathrm{mean}_I$
$\mathrm{cov}_{Ip} = \mathrm{corr}_{Ip} - \mathrm{mean}_I. * \mathrm{mean}_{Ip}$
$a = \mathrm{cov}_{Ip}./(\mathrm{var}_I + \epsilon)$
$b = \mathrm{mean}_p - a. * \mathrm{mean}_I$
3: $\mathrm{mean}_{a_H} = f_{RowSum}(a, r)$
$\mathrm{mean}_{b_H} = f_{RowSum}(b, r)$
4: $\mathrm{mean}_a = f_{ColumnSum}(\mathrm{mean}_{a_H}, r)$
$\mathrm{mean}_b = f_{ColumnSum}(\mathrm{mean}_{b_H}, r)$
$q = \mathrm{mean}_a. * I + \mathrm{mean}_b$

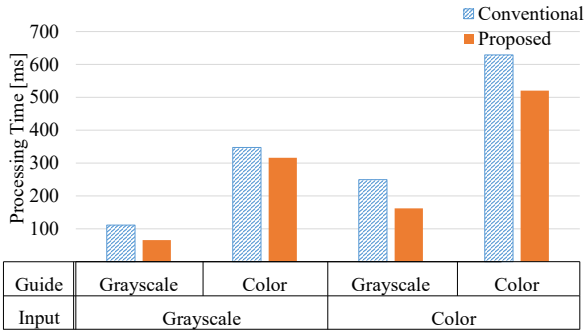

図 1: 実験結果．

ボックスフィルタと別の演算との結合を行う方法を提案した．

謝辞

本研究は，科研費 15K16023 によって行われた．

参考文献

[1] K. He, J. Shun, and X. Tang, “Guided image filtering,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 6, pp. 1397–1409, 2013.

[2] F. C. Crow, “Summed-area tables for texture mapping,” in *Proc. ACM SIGGRAPH*, 1984, pp. 207–212.

[3] M. McCool, J. Reinders, and A. Robison, *Structured Parallel Programming: Patterns for Efficient Computation*, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012.