

加重係数の多項式近似による セパラブルバイラテラルフィルタリングの高精度化

山下 頌太^{1,a)} 福嶋 慶繁^{1,b)}

概要: 本稿では、フィルタカーネルの空間分解によるバイラテラルフィルタの近似高速化手法について、より高精度な手法を提案する。従来の空間分解によるアプローチは、カラー画像に適用しても高速に動作するという利点があるが、近似精度が低い。この問題の改善のために、バイラテラルフィルタの空間分解による近似を定式化し、それをもとに、加重係数のマクローリン展開に基づいた高精度化手法を提案する。また、本手法をカラー画像に適用する場合、計算コストが急激に増加するという問題を抑制するための実装方法についても提案し、実験でその有効性を検証する。

1. はじめに

バイラテラルフィルタ (Bilateral Filter: BF) [1] は、画像の平滑化とエッジ保持を両立したフィルタであり、その優れたフィルタ特性から、ノイズ除去 [2]、ハイダイナミックレンジ画像 [3]、超解像 [4]、霞除去 [5]、テクスチャ強調 [6]、スタイライゼーション [7] といった、様々な画像処理アプリケーションに応用されている。このフィルタは、注目・参照画素間の空間距離と輝度差に基づいた適応的な重みを用いて処理を行う。この処理には高い計算コストを要するため、リアルタイムアプリケーションへの適用のために、様々な高速化手法が提案されている。

現在の主流となっている高速化アルゴリズムでは、BFを複数のガウシアンフィルタ処理に分解することで、フィルタカーネルの大きさに依存しない処理を実現している [8], [9], [10]。これらの手法は、グレースケール画像に対しては非常に有効だが、チャンネルの高次元化に伴う計算量の増加が著しく、カラー画像処理には向かない。カラー画像処理の効率化を目的とした定数時間アルゴリズム [11], [12] も提案されているが、これらは処理に用いるデータの構造が複雑であり、CPU 上でのベクトル演算との相性が悪いため、処理効率を高めるためには GPU での実装が必須である。

本稿で取り上げるセパラブルバイラテラルフィルタ (Separable Bilateral Filter: SBF) [13], [14] は、カーネルサイ

ズが中規模程度までであれば、カラー画像に対しても高速に動作する手法である。基本的な BF では、2 次元の畳み込み演算が行われるが、SBF では、フィルタカーネルを近似的に空間分解し、1 次元の畳み込み演算の組み合わせによって処理が達成される。メモリの消費量も少ないため、リアルタイム処理を要するアプリケーションにおいて有用だが、近似精度が低いことが問題である。文献 [15] により、近似精度を向上させるための実装方法が提案されているが、その精度向上では十分ではない。

本稿では、SBF の更なる高精度化のため、この空間分解による近似を定式化し、加重係数のマクローリン展開に基づいた高精度化手法を提案する。また、カラー画像に対して本手法を適用する場合、計算コストが急激に増加するという問題を抑制するための実装方法についても提案し、実験でその有効性を検証する。

2. 関連研究

2.1 バイラテラルフィルタ

入力画像を f とすると、BF の出力画像 \bar{f} は、以下の式で求められる。

$$\bar{f}_{i,j} = \frac{\sum_{x=-r}^r \sum_{y=-r}^r w_s(x,y)w_c(x,y)f_{i+x,j+y}}{\sum_{x=-r}^r \sum_{y=-r}^r w_s(x,y)w_c(x,y)} \quad (1)$$

ここで、 i, j は注目画素位置を表しており、 r はカーネル半径である。 w_s と w_c はそれぞれ注目・参照画素間の空間距離と輝度差から計算される加重係数であり、以下の式で定義される。

¹ 名古屋工業大学
Nagoya Institute of Technology
a) s.yamashita.171@stn.nitech.ac.jp
b) fukushima@nitech.ac.jp

$$w_s(x, y) = \exp\left(-\frac{x^2 + y^2}{2\sigma_s^2}\right) \quad (2)$$

$$w_c(x, y) = \exp\left(-\frac{\|\mathbf{f}_{i,j} - \mathbf{f}_{i+x,j+y}\|_2^2}{2\sigma_c^2}\right) \quad (3)$$

ここで、 σ_s と σ_c はそれぞれ空間平滑化とエッジ保持の度合いを決めるスケールパラメータである。

2.2 セパラブルバイラテラルフィルタ

2次元空間フィルタを FIR 形式で単純に実装した場合、その計算量は $O(r^2)$ であり、このカーネル半径に対する計算コストの非線形な増加は、フィルタ処理の応用に大きな制約を与える。セパラブル実装は、この問題を解決するための古典的なアプローチであり、フィルタカーネルを1次元の垂直・水平フィルタに分解することで、計算量を $O(r)$ へと削減することができる。カーネルが可分性を持つフィルタでは、この手法によって近似なしでの高速化が可能だが、BF は w_c が可分性を持たないため、出力は近似値であり、近似精度についての議論が必要である。また、BF のセパラブル実装はフィルタリング方向の順序によって出力が異なるが、以降では、1パス目の処理が垂直方向、2パス目の処理が水平方向であると仮定して説明する。

SBF は、BF のセパラブル実装であり、最初に Pham らによって提案された [13]。その実装では、まず、入力画像に対して垂直方向に BF を適用し、中間画像を生成する。その後、中間画像に対して水平方向に BF を適用し、最終的な出力を得る。なお、水平方向の BF の重みは中間画像の輝度値を参照して計算される。

Pham らの実装方法は、高速かつ省メモリだが、近似精度が低いことが問題として挙げられる。この問題に対して、水平方向フィルタに用いる重みを、中間画像からではなく、入力画像を参照して計算することで精度を向上させる手法が提案されている [15]。中間画像と比較して、入力画像は正しいテクスチャ情報を持っていることから、この手法を適用することで、近似精度が有意に向上する。また、これに加えて文献 [15] では、SBF の出力に平滑化効果が過剰にかかりやすいという傾向に着目し、平滑化効果を抑えるためのパラメータを水平方向フィルタの計算式に追加することで近似誤差を減らしている。この手法はスイッチングデュアルカーネル (Switching Dual Kernels: SDK) と呼ばれ、本稿では SDK を SBF に適用した実装を SDK-SBF と略記する。初期の SBF に比べると、上記の手法により大幅な高精度化がされているが、それでもなお、十分な精度の出力が得られているとは言えず、より高精度な出力を得るための改善案が求められる。

3. セパラブルバイラテラルフィルタの定式化

ここでは、SBF による BF の近似を定式化し、より高精度な出力を得るアプローチを提案する。なお、簡単のため、画像はグレースケールであるとする。まず、BF のフィル

タカーネルが x, y 方向に分解可能であると仮定し、二種類の重み w_s, w_c を空間分解しやすい形に式変形する。 w_s は次式のように変形する。

$$w_s(x, y) = w_s^H(x)w_s^V(y) \quad (4)$$

$$w_s^H(x) = \exp\left(-\frac{x^2}{2\sigma_s^2}\right) \quad (5)$$

$$w_s^V(y) = \exp\left(-\frac{y^2}{2\sigma_s^2}\right) \quad (6)$$

また、 w_c は次式のように変形する。

$$w_c(x, y) = w_c^H(x)w_c^V(y)w_c^N(x, y) \quad (7)$$

$$w_c^H(x) = \exp(-\gamma D_{i,j}^2(x)) \quad (8)$$

$$w_c^V(y) = \exp(-\gamma D_{i+x,j}^2(y)) \quad (9)$$

$$w_c^N(x, y) = \exp(-2\gamma D_{i,j}(x)D_{i+x,j}(y)) \quad (10)$$

ここで、 $\gamma = \frac{1}{2\sigma_c^2}$ であり、 $D_{i,j}(x) = f_{i,j} - f_{i+x,j}$ 、 $D_{i+x,j}(y) = f_{i+x,j} - f_{i+x,j+y}$ である。次に、式 (4-10) を式 (1) に代入し、整理すると以下の式になる。

$$\bar{f}_{i,j} = \frac{\sum_{x=-r}^r w_s^H(x)w_c^H(x) \sum_{y=-r}^r w_s^V(y)w_c^V(y)w_c^N(x, y)f_{i+x,j+y}}{\sum_{x=-r}^r w_s^H(x)w_c^H(x) \sum_{y=-r}^r w_s^V(y)w_c^V(y)w_c^N(x, y)} \quad (11)$$

式 (11) において、 w_c^N だけが x, y 方向に分解できず、それ以外は分解可能になっている。そのため、 w_c^N を x, y 方向に分解可能な近似式で置き換えることが出来れば、BF は分解可能となる。以下では、この w_c^N の置き換えについて、3通りの方法を取り上げ、従来の SBF との関係について述べる。

定数による置き換え

最も単純な方法は、 $w_c^N = 1$ とした、定数での置き換えである。この置き換えを行った場合は、2.2 節で紹介した、水平方向フィルタ時に入力画像を参照する実装と類似した処理内容となり、違いは垂直方向フィルタ後の正規化処理の有無だけである。

$D_{i,j}(x)$ のみの関数による置き換え

$D_{i,j}(x)$ と $D_{i+x,j}(y)$ に線形の関係があると仮定し、 w_c^N を $D_{i,j}(x)$ のみの関数で置き換えることによって、BF を可分にしつつ、SDK-SBF の式を導出することができる。まず、定数 t を用いて、 $D_{i+x,j}(y) = tD_{i,j}(x)$ とし、これを式 (10) に代入したものを $w_c'^N$ とする。

$$w_c'^N(x, y) = \exp(-2\gamma t D_{i,j}^2(x)) \quad (12)$$

そして、指数関数を用いて w_c^H と $w_c'^N$ を統合することで、 $w_c'^N(x, y)$ の項が消え、SDK-SBF [15] の水平方向フィルタで用いられる重み w_c^H と同様の重みを得られる。

$$\begin{aligned} w_c'^H &= \exp(-\gamma D_{i,j}^2(x) - 2\gamma t D_{i,j}^2(x)) \\ &= \exp(-\frac{1}{\alpha^2} \gamma D_{i,j}^2(x)) \end{aligned} \quad (13)$$

ここで、 $\alpha = \frac{1}{\sqrt{1+2t}}$ であり、平滑化効果を抑制するためのハイパーパラメータとして扱われている [15]. α を $[0, 1]$ の区間で指定することで、 w_c^N を定数で置き換える場合と比べて、近似誤差を減らすことができる。

マクローリン級数による置き換え

更なる高精度化のため、 w_c^N をマクローリン展開することで得られる級数を近似に用いる手法を提案する。 w_c^N のマクローリン展開は、次式で表すことができる。

$$w_c^N(x, y) = \sum_{k=0}^{\infty} \frac{(-2\gamma D_{i,j}(x) D_{i+x,j}(y))^k}{k!} \quad (14)$$

式 (14) における級数の各項は、 x, y 方向に計算を分解可能である。また、指数関数のマクローリン展開の収束半径は ∞ であるため、式 (14) を式 (11) に代入することで、BF を分解可能にしつつ、近似なしの出力を得ることが出来る。高速化にあたり、式 (14) の級数を低次の項で打ち切る必要があるが、前述した定数による w_c^N の置き換えが、級数を第 1 項で打ち切った場合と等しいことから、級数の第 2 項以降を計算に用いることで、従来の実装よりも高精度な出力を得ることが期待できる。以下では、級数を第 1 項で打ち切ったものを 0 次近似 SBF (0-SBF)、第 2 項で打ち切ったものを 1 次近似 SBF (1-SBF) と呼ぶ。なお、2 次近似以降は計算コストの増加に対する精度向上が小さかったため、本稿では扱わない。

4. 1-SBF の実装方法

ここでは、1-SBF の具体的な実装方法について述べる。なお、入力はグレースケール画像であるとする。実装にあたり、新たに $D_{i,j}(y) = f_{i,j} - f_{i,j+y}$ とおき、式 (9) で示した w_c^V を $w_c^V(y) = \exp(-\gamma D_{i,j}^2(y))$ と再定義する。1-SBF では、分子と分母がそれぞれ 2 つの項で構成されるため、計 4 枚の中間画像を生成する必要がある。分子を構成する項の中間画像を $g_{i,j}^{(0)}$ と $g_{i,j}^{(1)}$ とし、分母を構成する項の中間画像を $h_{i,j}^{(0)}$ と $h_{i,j}^{(1)}$ とすると、次式に示す垂直方向フィルタ処理によって、それぞれが生成される。

$$g_{i,j}^{(0)} = \sum_{y=-r}^r w_s^V(y) w_c^V(y) f_{i,j+y} \quad (15)$$

$$g_{i,j}^{(1)} = \sum_{y=-r}^r w_s^V(y) w_c^V(y) D_{i,j}(y) f_{i,j+y} \quad (16)$$

$$h_{i,j}^{(0)} = \sum_{y=-r}^r w_s^V(y) w_c^V(y) \quad (17)$$

$$h_{i,j}^{(1)} = \sum_{y=-r}^r w_s^V(y) w_c^V(y) D_{i,j}(y) \quad (18)$$

その後、これらの中間画像に対して水平方向フィルタ処理を行い、以下のように分子 $p_{i,j}$ と分母 $q_{i,j}$ を計算する。

$$p_{i,j} = \sum_{x=-r}^r w_s^H(x) w_c^H(x) (g_{i+x,j}^{(0)} - 2\gamma D_{i,j}(x) g_{i+x,j}^{(1)}) \quad (19)$$

$$q_{i,j} = \sum_{x=-r}^r w_s^H(x) w_c^H(x) (h_{i+x,j}^{(0)} - 2\gamma D_{i,j}(x) h_{i+x,j}^{(1)}) \quad (20)$$

ここで、 $g_{i+x,j}^{(1)}$ と $h_{i+x,j}^{(1)}$ は、以下の閾値処理によって得られる値である。

$$g_{i+x,j}^{(1)} = \begin{cases} g_{i+x,j}^{(1)} & h_{i+x,j}^{(0)} - 2\gamma D_{i,j}(x) h_{i+x,j}^{(1)} > 0 \\ 0 & otherwise \end{cases} \quad (21)$$

$$h_{i+x,j}^{(1)} = \begin{cases} h_{i+x,j}^{(1)} & h_{i+x,j}^{(0)} - 2\gamma D_{i,j}(x) h_{i+x,j}^{(1)} > 0 \\ 0 & otherwise \end{cases} \quad (22)$$

この閾値処理で行っていることは、1-SBF から 0-SBF への処理の切り替えである。1-SBF では指数関数を 1 次関数で近似しており、加重係数として 0 以下の値が発生する可能性がある。加重平均の定義上、重みとして負数を用いることは極力避けるべきであり、重みが全て 0 になる可能性も排除すべきである。そこで、式 (21)、式 (22) のように、 $q_{i,j}$ に加算される数値の符号をチェックし、正の値になっていなければ、0-SBF の重みを用いるようにする。0-SBF では 0 以下の重みが発生しないため、この処理の切り替えにより、出力の安定性が向上する。最後に、 $p_{i,j}$ と $q_{i,j}$ の比を取ることで、出力を得る。

$$\bar{f}_{i,j} = \frac{p_{i,j}}{q_{i,j}} \quad (23)$$

なお、式 (16) と式 (18) において、 $g_{i,j}^{(1)} = 0$ 、 $h_{i,j}^{(1)} = 0$ とすることで、0-SBF のアルゴリズムとなる。

5. カラー画像処理への拡張

5.1 3チャンネルの加重係数に対するマクローリン展開

式 (14) で示した加重係数のマクローリン展開を 3 チャンネルのカラー画像処理に拡張する方法を 2 通り示す。

まず 1 つ目は、式 (14) を、各チャンネルの指数関数に対して独立に適用する方法である。入力の RGB 信号をそれぞれ、 $f_{i,j}^{(R)}$ 、 $f_{i,j}^{(G)}$ 、 $f_{i,j}^{(B)}$ とし、3 章の時と同じ要領で、各チャンネルに対して $D_{i,j}(x)$ と $D_{i+x,j}(y)$ を定義すると、この級数展開は次式で表せる。

$$w_c^N(x, y) = \prod_{m \in \omega} \sum_{k=0}^{\infty} \frac{(-2\gamma D_{i,j}^{(m)}(x) D_{i+x,j}^{(m)}(y))^k}{k!} \quad (24)$$

ここで、 ω はチャンネルの集合である。近似の次数を k とした時、この展開方法で必要となる項数は、 $(k+1)^3$ 項であり、1 次近似の場合は 8 項となる。

2つ目は、指数法則を用いて3チャンネルの指数関数を統合し、それを1つの指数関数と見なして展開を行う方法である。これは、次式で表せる。

$$w_c^N(x, y) = \sum_{k=0}^{\infty} \frac{(-2\gamma \sum_{m \in \omega} D_{i,j}^{(m)}(x) D_{i+x,j}^{(m)}(y))^k}{k!} \quad (25)$$

この展開方法で必要となる項数は、 $\frac{3^{k+1}-1}{2}$ 項であり、1次近似の場合は4項となる。

5.2 1チャンネルジョイント信号による1次近似

前節で示した展開方法では、1次近似であっても、0次近似の約8倍、あるいは約4倍の計算コストを要する。この項数の急激な増加を緩和するため、入力 RGB 信号を1チャンネル信号へと変換し、それを w_c^N の近似に代用する手法を提案する。本手法では、1次近似に要する項数は2項である。

信号の変換には単純平均法を用いる。変換後の1チャンネル信号を $f_{i,j}^{(Y)}$ とすると、この変換は次式で表される。

$$f_{i,j}^{(Y)} = \frac{f_{i,j}^{(R)} + f_{i,j}^{(G)} + f_{i,j}^{(B)}}{3} \quad (26)$$

$f_{i,j}^{(Y)}$ について、これまでと同様に $D_{i,j}(x)$ と $D_{i+x,j}(y)$ を定義し、この信号を式 (25) に代用する。これは次式で表される。

$$w_c^N(x, y) \approx \sum_{k=0}^1 \frac{(-2\gamma \cdot 3D_{i,j}^{(Y)}(x)D_{i+x,j}^{(Y)}(y))^k}{k!} \quad (27)$$

1チャンネル信号に変換したことで、色の情報は落ちるが、画像中のおおよそのテクスチャ情報は残っているため、本手法により、0-SBF 以上の近似精度を得ることが期待できる。また、 w_c^H と w_c^V については、計算に RGB 信号を用いており、 $w_c^N(x, y)$ のみ式 (26) を用いていることに注意されたい。

6. 実験および考察

本章では、提案した1-SBFの計算時間と近似精度について、まずは従来の SBF との比較を行い、その後 SBF 以外的高速化手法との性能比較を行う。各実験において、SDK-SBF で用いる α の値は 0.8 とし、これは文献 [15] における α の最適値である。実験で扱う各手法は C++ で実装されており、SIMD 命令、OpenMP を用いてベクトル化、並列化されている。また、CPU として Intel Core i7-6700 3.40GHz を用いている。

6.1 計算時間

サイズ 1024×1024 のカラー画像に対してフィルタリング処理を行った際の計算時間を図 1 に示す。フィルタカーネル半径 r は、 $3\sigma_s$ とした。なお、SDK-SBF は 0-SBF と処理時間がほぼ同じであるため、図からは省略した。ま

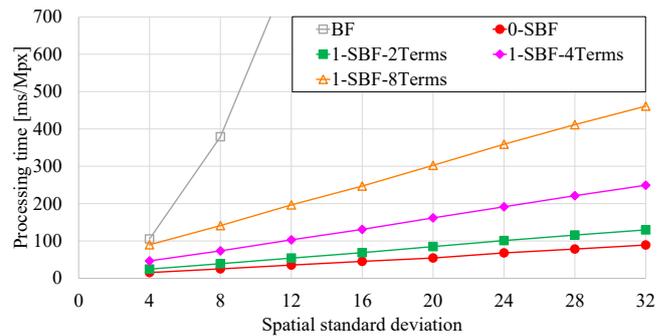


図 1: SBF の各実装の処理時間。

た、参考として、グレースケール処理において最も効率の良い定数時間アルゴリズムであるリアルタイム $O(1)$ バイラテラルフィルタ [9] やコンプレッションバイラテラルフィルタ [10] は、カラー画像での処理効率は低く、特に σ_c が小さい場合はこの処理に 10000ms 以上を要する。

いずれの SBF 実装も、処理時間が線形に増加しており、カーネルサイズへの依存性を抑制できていることがわかる。また、近似に用いる項数と処理時間の関係もほぼ線形であり、1-SBF に1チャンネル信号を用いることで、計算コストの大幅な削減がされている。

6.2 近似精度

近似精度の客観評価尺度には MSE を使い、正解は2次元カーネルによる BF の出力とする。データセットとして、Kodak 社により提供されている 24 枚のカラー画像を用い、それらの MSE の平均値によって比較を行う。

比較結果を図 2 に示す。これらの図において、縦軸は対数軸であることに注意されたい。1-SBF はいずれの実装方法でも、SDK-SBF 以上に高精度な近似が出来ている。特に、2項による 1-SBF の近似精度は、4項による 1-SBF とほぼ同程度であり、マクロローリン展開式に1チャンネル信号を代用しても近似誤差を大幅に減らせることがわかる。また、1-SBF 全般の傾向として、 σ_c が大きいほど近似精度が高くなっている。これは、多項式近似がマクロローリン展開に基づいたものであり、指数の引数が 0 に近いほど近似が成立しやすくなるためであると考えられる。

視覚的な評価のため、図 3 に各手法の処理結果を示す。BF の処理結果に比べ、0-SBF は平滑化が過剰にかかり、細かいテクスチャが失われてしまっているが、SDK-SBF と 1-SBF ではそれを抑制出来ている。

6.3 他の高速化手法との比較

SBF 以外の近似高速化手法と計算効率を比較し、提案手法の性能を検証する。処理時間はサイズ 1024×1024 のカラー画像への処理に要する時間で評価し、近似精度は Kodak 社より提供されているカラー画像 24 枚における平均精度で評価する。なお、この実験では、精度の客観評価尺度として、SSIM [16] を用いている。比較手法として、乱択バイラテラルフィルタ (Randomized Bilateral Filter:

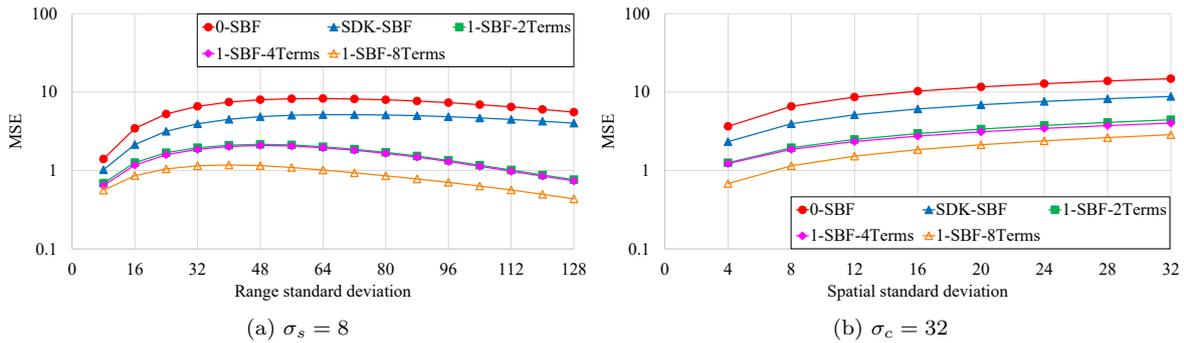


図 2: SBF の各実装の近似精度.

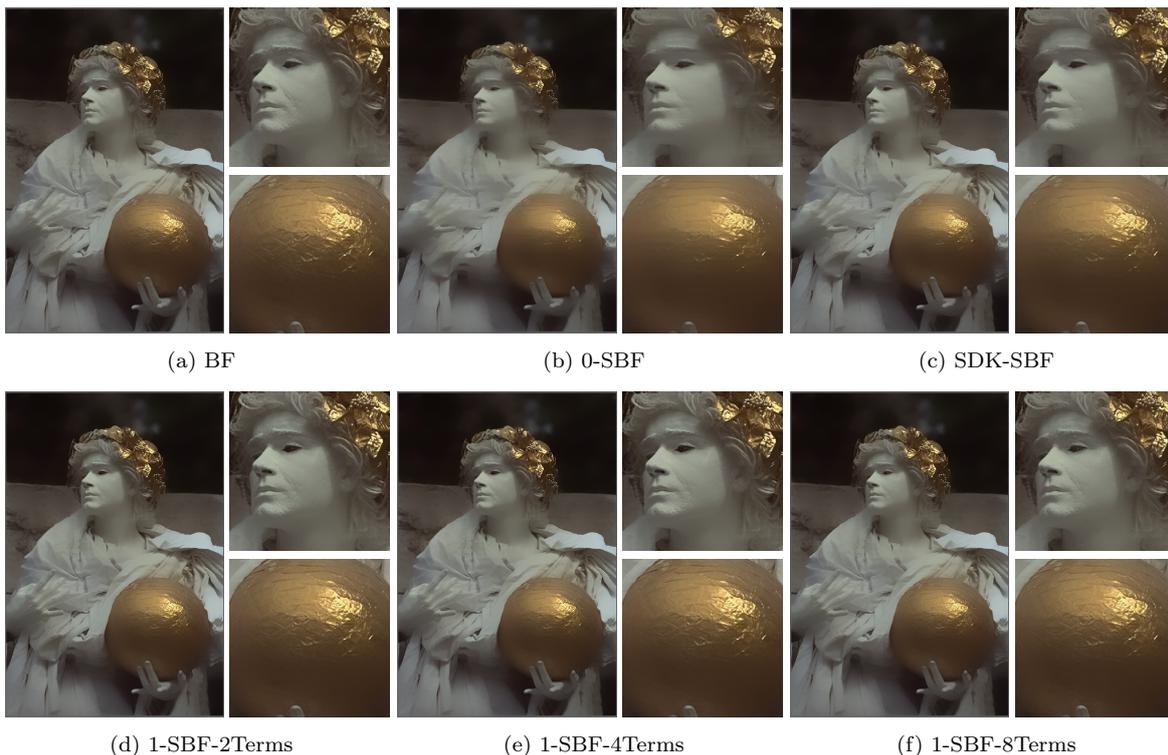


図 3: バイラテラルフィルタリングの結果. 図 (b), (c), (d), (e), (f) の近似精度はそれぞれ, 37.75 dB, 40.27 dB, 44.38 dB, 44.45 dB, 47.08 dB である. なお, 使用した平滑化パラメータは $\sigma_s = 16$, $\sigma_c = 48$, カーネル半径は $3\sigma_s$ である.

RBF) [17] を用いる. RBF は, BF のカーネルが持つ情報の冗長性を利用し, カーネル内の画素をランダムに間引くことで計算コストを削減する手法であり, SBF と同様に, カラー画像に対して十分高速な処理を行うことができる. RBF は間引き率を変化させることで計算コストと精度のバランスを柔軟に取ることが可能であり, 実験では, 提案した SBF とほぼ同等の計算コストになるよう, 間引き率を調整している.

図 4 に比較結果を示す. 参考として, 最初期の SBF である文献 [13] の実装の結果も記載した. SBF はカーネルサイズが大きいほど近似の効率が落ちやすいため, σ_s が大きくなるほど RBF に比べて処理効率が悪くなるが, σ_s が小さい時は, RBF よりも高効率な処理ができてい

がわかる. 8 項による 1-SBF は処理時間が大きく, 精度とのバランスが取れていないが, 2 項による 1-SBF や σ_c が大きい場合の 4 項による 1-SBF は, RBF と比較して良好な処理効率を実現している. また, RBF は σ_c が大きいほど処理効率が落ちるという性質があるが, 1-SBF は σ_c が大きいほど近似精度が高くなるため, 平滑化パラメータに応じてこれらを使い分けることで, アプリケーションの性能向上が期待出来る.

7. まとめ

本稿では, フィルタカーネルの空間分解によるバイラテラルフィルタの近似高速化手法を定式化し, その更なる高精度化に向け, 加重係数のマクローリン展開に基づく実装

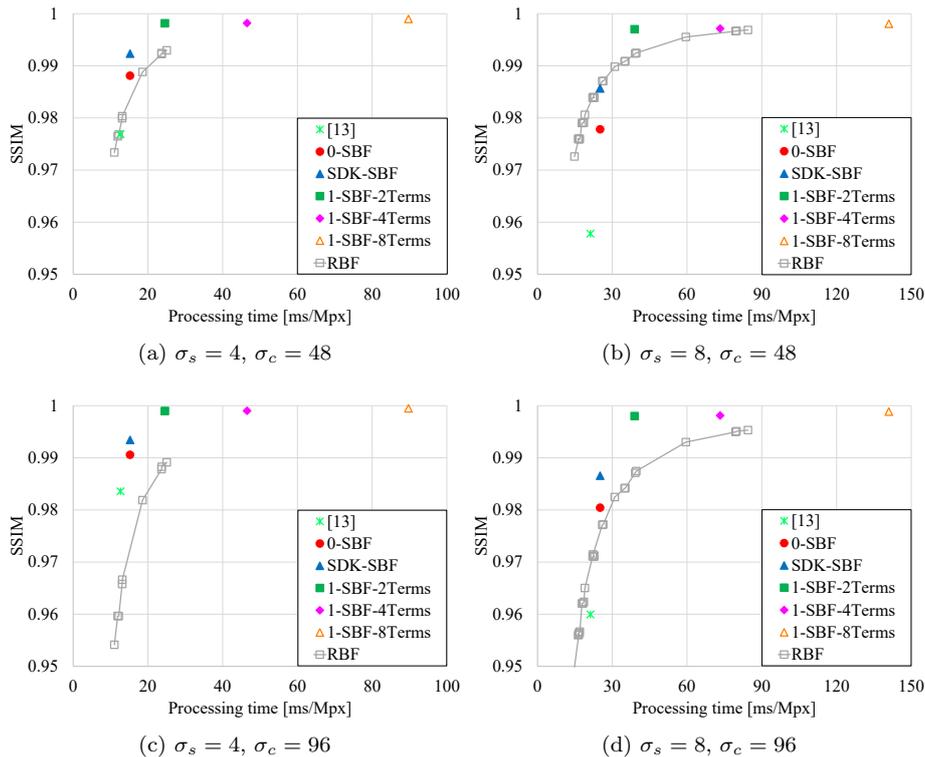


図 4: SBF の各実装と RBF の比較。

方法を提案した。また、級数展開をカラー画像処理に拡張した際の問題として生じる項数の急激な増加に対して、1チャンネルの信号を近似式に代用することで処理効率を向上させる手法を提案し、実験でその有効性を示した。

謝辞 本研究は科研費 JP17H01764 の助成を受けた。

参考文献

- [1] Tomasi, C. and Manduchi, R.: Bilateral Filtering for Gray and Color Images, *IEEE International Conference on Computer Vision (ICCV)*, pp. 839–846 (1998).
- [2] Buades, A., Coll, B. and Morel, J. M.: A Non-Local Algorithm for Image Denoising, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 60–65 (2005).
- [3] Durand, F. and Dorsey, J.: Fast Bilateral Filtering for the Display of High-Dynamic-Range Images, *ACM Trans. on Graphics*, Vol. 21, No. 3, pp. 257–266 (2002).
- [4] Kopf, J., Cohen, M. F., Lischinski, D. and Uyttendaele, M.: Joint Bilateral Upsampling, *ACM Trans. on Graphics*, Vol. 26, No. 3, pp. 96:1–96:5 (2007).
- [5] He, K., Sun, J. and Tang, X.: Single Image Haze Removal Using Dark Channel Prior, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 33, No. 12, pp. 2341–2353 (2011).
- [6] Bae, S., Paris, S. and Durand, F.: Two-scale Tone Management for Photographic Look, *ACM Trans. on Graphics*, Vol. 25, No. 3, pp. 637–645 (2006).
- [7] DeCarlo, D. and Santella, A.: Stylization and Abstraction of Photographs, *ACM Trans. on Graphics*, Vol. 21, No. 3, pp. 769–776 (2002).
- [8] Porikli, F.: Constant Time O(1) Bilateral Filtering, *IEEE Conference on Computer Vision and Pattern*

- Recognition (CVPR)*, pp. 1–8 (2008).
- [9] Yang, Q., Tan, K. H. and Ahuja, N.: Real-Time O(1) Bilateral Filtering, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 557–564 (2009).
- [10] Sugimoto, K. and Kamata, S.-I.: Compressive Bilateral Filtering, *IEEE Trans. on Image Processing*, Vol. 24, No. 11, pp. 3357–3369 (2015).
- [11] Adams, A., Gelfand, N., Dolson, J. and Levoy, M.: Gaussian KD-trees for Fast High-dimensional Filtering, *ACM Trans. on Graphics*, Vol. 28, No. 3, pp. 21:1–21:12 (2009).
- [12] Adams, A., Baek, J. and Davis, M. A.: Fast High-Dimensional Filtering Using the Permutohedral Lattice, *Computer Graphics Forum*, Vol. 29, No. 2, pp. 753–762 (2010).
- [13] Pham, T. Q. and Vliet, L. J. V.: Separable Bilateral Filtering for Fast Video Preprocessing, *IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–4 (2005).
- [14] Kim, Y. S., L., H., Choi, O., Lee, K., Kim, J. and Kim, J.: Separable bilateral nonlocal means, *IEEE International Conference on Image Processing (ICIP)*, pp. 1513–1516 (2011).
- [15] Fukushima, N., Fujita, S. and Ishibashi, Y.: Switching Dual Kernels for Separable Edge-Preserving Filtering, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1588–1592 (2015).
- [16] Wang, Z., Bovik, A., Sheikh, H. and Simoncelli, E.: Image Quality Assessment: from Error Visibility to Structural Similarity, *IEEE Trans. on Image Processing*, Vol. 13, No. 4, pp. 600–612 (2004).
- [17] Banterle, F., Corsini, M., Cignoni, P. and Scopigno, R.: A Low-Memory, Straightforward and Fast Bilateral Filter Through Subsampling in Spatial Domain, *Computer Graphics Forum*, Vol. 31, No. 1, pp. 19–32 (2012).