Vol.2020-CVIM-220 No.18
2020/1/24

Local Trilateral Upsampling の GPU 並列実装による高速化

門脇奨[†1]　和田俊和[†2]

概要：遠赤外線カメラや Time of Fright(ToF)型深度カメラは可視光カメラに比べ，価格が高く，撮影される画像の解像度は高くないという問題がある．このような問題を解決するために A. Cvetkovic らは Local Trilateral Upsampling(LTU)という手法を提案している．しかし，LTU は輝度値に依存する shift-variant な画像フィルタであり，フィルタの重みのパラメータを局所相互情報量(Local Mutual Information:LMI)によって動的に変更するため計算時間が長いという問題がある．本研究では， Bilateral Filter の高速化手法である Permutohedral Lattice を用いて，LTU を GPU で並列実装することで高速化を行う．また，LMI はガイド画像のエッジ付近で強い値が出るため，エッジ情報で代替可能である．そのため，本研究では LMI の代わりにエッジ情報を用いる手法について検討し，これを用いた LTU の GPU 並列実装も行う．

キーワード：Local Trilateral Upsampling，クロスモーダル超解像処理，GPU を用いた並列化，温度画像,

Acceleration by GPU Parallelization for Local Trilateral Upsampling

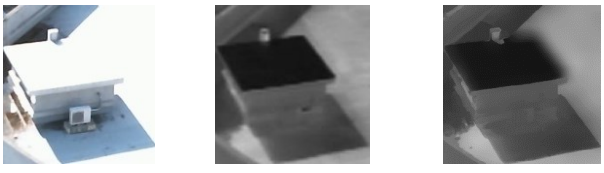
SUSUMU KADOWAKI[†1]　TOSHIKAZU WADA[†2]

1. はじめに

自動運転で道路上の歩行者や動物を検出するために有効であるとされているマイクロボロメータ型遠赤外線カメラによる画像や Time of Fright(ToF)型深度カメラによる画像は，可視光カメラによる画像よりも解像度が低いうえにノイズが多い．また，カメラ自体の値段も VGA(640 × 480)画質の画像を撮影するマイクロボロメータ型遠赤外線カメラが約 90 万円程度と Full-HD(1920 × 1080)画質の可視光カメラが約 1 万円で購入できることと比べると高価であるという問題がある．このような問題は，Joint Bilateral Upsampling(JBU)に代表されるエッジ保持平滑化フィルタによる高解像度化で解決することが出来る．

1.1 JBU の問題点

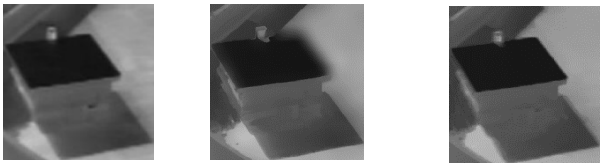
図 1 は a)の可視光画像をガイド画像とし，b)の温度画像をターゲット画像として高解像度化した例であり，c)の JBU 結果画像では一部のエッジがボケていることが分かる．この例のように JBU には，ターゲット画像にあるエッジがガイド画像に存在しない場合に高解像度化に失敗するという問題がある．



a) 可視光画像　b) 温度画像　c) JBU 結果画像

図 1 JBU の計算が破綻する例

1.2 Trilateral Upsampling[1]



a) 温度画像　b) JBU 結果画像　c) TU 結果画像

図 2 JBU と TU の結果比較

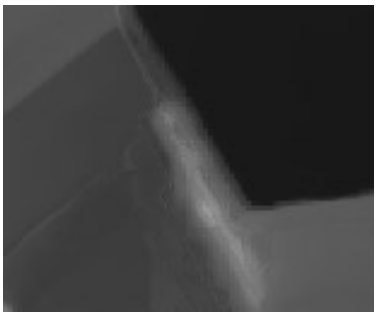


図 3 TU で発生するジャギーの例

A. Cvetkovic ら[1]は JBU の計算が破綻する問題を解決する方法として Trilateral Upsampling(TU)を提案している．

TU は，点$\boldsymbol{p}$における高解像度化の対象画像の輝度を$T_{\boldsymbol{p}}$，ガイド画像の輝度を$\boldsymbol{I}_{\boldsymbol{p}}$とした時，式(1)で表される．

$$T_{\boldsymbol{p}}^{TU} = \frac{1}{W_{\boldsymbol{p}}^{TU}} \sum_{\boldsymbol{q}\in S} \mathrm{G}_{\sigma_s}(\|\boldsymbol{p}-\boldsymbol{q}\|)\mathrm{G}_{\sigma_I}(\|\boldsymbol{I}_{\boldsymbol{p}}-\boldsymbol{I}_{\boldsymbol{q}}\|)\mathrm{G}_{\sigma_T}(|T_{\boldsymbol{p}}-T_{\boldsymbol{q}}|)T_{\mathbf{q}} \quad (1)$$

1 和歌山大学　大学院　システム工学研究科　システム工学専攻
Wakayama Univ. Graduate School of System Engineering
2 和歌山大学　システム工学部
Wakayama Univ. Faculty of System Engineering

ただし，

$$G_{\sigma_y}(x) = \exp(-\frac{x^2}{2(\sigma_y)^2})$$

$$W_p^{TU} = \sum_{q \in S} G_{\sigma_s}(\|\boldsymbol{p}-\boldsymbol{q}\|) G_{\sigma_I}(\|\boldsymbol{I_p}-\boldsymbol{I_q}\|) G_{\sigma_T}(|T_p - T_q|)$$

TU は，JBU とは違い，ガイド画像の輝度による重みだけでなく，高解像度化の対象画像の輝度による重みも利用しているため，図 1 のように JBU ではエッジがボケる場合であっても，図 2 のようにこの現象は起きない．

しかし，TU では，解像度の低い対象画像の影響を強く受けすぎてしまうため，図 3 のようにエッジ付近にジャギーと呼ばれる階段状のノイズが発生する問題がある．

1.3 Local Trilateral Upsampling[1]

文献[1]では，この問題を解決した Local Trilateral Upsampling(LTU)と呼ばれる手法も提案されている．

LTU は，式(2)で定義される局所相互情報量(Local Mutual Information:LMI)を用いて評価した「対象画像とガイド画像の局所領域での輝度の一貫性」に基づいて TU の重み付けに用いられているパラメータσ_R，σ_G，σ_B，σ_Tを，以下に示すように動的に変化させる方法である．

ここで，画像間の局所相互情報量は，

$$\mu_{IT} = \sum_{I \in S}\sum_{T \in S} h_{IT}(I,T)\log(\frac{h_{IT}(I,T)}{h_I(I)h_T(T)}) \quad (2)$$

と表される．ただし，$h_x(I)$は画像X中の小領域S内の輝度のヒストグラム，$h_{XY}(I,T)$は，画像X，Yの小領域S内の輝度の 2 次元ヒストグラムを表す．

LTU の計算式は式(3)で表される．TU の場合ガイド画像となる可視光画像の輝度を用いた重みはチャネルに関わらず同じ重みを使用していたが，LTU ではチャネルごとに異なる重みを用いる．このとき，$W_p^{I_R}$，$W_p^{I_G}$，$W_p^{I_B}$，W_p^{T}のパラメータσ_R，σ_G，σ_B，σ_Tは，式(4)～(7)で表される．

このようにして計算された LTU は，図 4 のようにジャギーが改善されている．

$$T_p^{LTU} = \frac{1}{W_p^{LTU}}\sum_{q \in S} W_p^S W_p^{I_R} W_p^{I_G} W_p^{I_B} W_p^T T_q \quad (3)$$

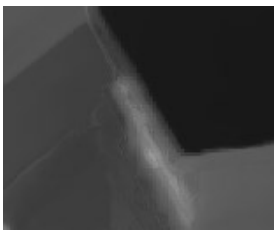
ただし，

$$W_p^S = \exp(-\frac{(\|\boldsymbol{p}-\boldsymbol{q}\|)^2}{2(\sigma_s)^2}) \ ,\ W_p^{I_R} = \exp(-\frac{(|R_p - R_q|)^2}{2(\sigma_R)^2})$$

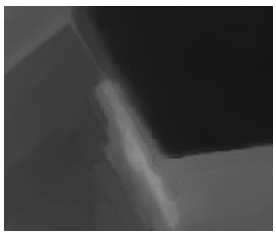
$$W_p^{I_G} = \exp(-\frac{(|G_p - G_q|)^2}{2(\sigma_G)^2}) \ ,\ W_p^{I_B} = \exp\left(-\frac{(|B_p - B_q|)^2}{2(\sigma_B)^2}\right)$$

$$W_p^T = \exp(-\frac{(|T_p - T_q|)^2}{2(\sigma_T)^2})$$

$$W_p^{LTU} = \sum_{q \in S} W_p^S W_p^{I_R} W_p^{I_G} W_p^{I_B} W_p^T$$



a)TU の例



b)LTU の例

図 4 TU と LTU の比較図

$$\sigma_R = \sigma\sqrt{\frac{1+(\mu_{RT}(\boldsymbol{p}))^2}{\mu_{RT}(\boldsymbol{p})}} \quad (4)$$

$$\sigma_G = \sigma\sqrt{\frac{1+(\mu_{GT}(\boldsymbol{p}))^2}{\mu_{GT}(\boldsymbol{p})}} \quad (5)$$

$$\sigma_B = \sigma\sqrt{\frac{1+(\mu_{BT}(\boldsymbol{p}))^2}{\mu_{BT}(\boldsymbol{p})}} \quad (6)$$

$$\sigma_T = \sqrt{\frac{1}{\frac{1}{\left(\sigma\sqrt{1+(\mu_{RT}(\boldsymbol{p}))^2}\right)^2}+\frac{1}{\left(\sigma\sqrt{1+(\mu_{GT}(\boldsymbol{p}))^2}\right)^2}+\frac{1}{\left(\sigma\sqrt{1+(\mu_{BT}(\boldsymbol{p}))^2}\right)^2}}} \quad (7)$$

これらの調整は，μ_{IT}が大きく，対象画像とガイド画像の局所領域での輝度の一貫性がある場合に JBU に近づき，一貫性が崩れた場合には TU に近づくように設計されている．

1.4 本研究の目的

LTU は，TU の計算を LMI に応じてパラメータを動的に変更しながら計算する必要があるため，TU より多くの計算時間がかかる．したがって，自動運転のように瞬時に高解像度画像が必要になるような用途で用いることは出来ない．

このため，本研究では， GPU を用いた並列実装によって LTU を高速化することを目的とする．

2. 関連研究

2.1 Trilateral Grid[2]

Trilateral Grid(TG)は，Bilateral Filter の高速化手法である Bilateral Grid[3]を TU に適用した手法である．これは，画像を多次元空間内の曲面として表現することにより，shift-variant なフィルタリングを，shift-invariant なフィルタリングとして実行する手法である．

TU をグリッド作成，平滑化，スライシングの 3 つの手順に分け，それぞれ GPU を用いて並列実装したものを順に実行することで単一 CPU よりも高速に実行することができる．

2.1.1 グリッド作成

グリッド作成では，画像上の空間x,yとガイド画像の輝度r,g,b，対象画像の輝度iの 6 つの軸を持つ空間中で対象画像を曲面として表現する．このとき，数式(1)の分母と分子

を別に保持しておき，後に説明するスライシングの段階でこれらの除算を行い結果画像を得ることが出来る．しかし，画像を高次元空間中で曲面として表現するため，データが入っていない配列要素が多く，無駄な計算を行う必要がある．そのため，空間を一定サイズの格子状に区切り，その格子内の配列要素の総和を保持するダウンサンプリングと呼ばれる処理を行うことで計算量を減らす．

2.1.2 平滑化

6 次元空間中で曲面として表現された対象画像を shift-invariant な 6 次元ガウシアンを用いて平滑化する．対象画像とガイド画像それぞれの輝度を軸として持っているため，ガウシアンを用いた平滑化を行っても「輝度値が大きく変化する対象画像のエッジ」をぼかしてしまうことは無い．

6 次元空間中で表現された画像は前述の通りスパースであるため，空間内のすべての点で平滑化計算を行うと元の画像で平滑化計算を行うよりも却って計算量が多くなってしまうことがある．そのため，フィルタ中心に輝度が存在する場合のみ平滑化計算を行うことで無駄な処理を省く．この処理は，フィルタ中心に輝度が存在せずフィルタ中心以外に輝度が存在する場合の計算も省略してしまうため，計算結果に若干の矛盾が生じるが，フィルタ中心以外に存在する輝度の影響度はフィルタ中心に存在する輝度の影響度に比べて小さいため結果画像に大きな矛盾は発生しない．

2.1.3 スライシング

グリッド作成の逆演算を行うことで，分母と分子のデータをそれぞれ取り出し，分子を分母で割ることで Trilateral Upsampling の計算結果画像を復元する処理．

グリッド作成の段階でダウンサンプリングを行っている場合，線形補間を用いてデータを補完し，画像を復元する．

2.1.4 Trilateral Grid の問題点

Trilateral Grid は，対象画像を高次元空間中の曲面として表現しているため平滑化計算を行う空間が広大になってしまい計算に時間がかかってしまう．そのため，自動運転のような高解像度の温度画像が瞬時に必要になるような用途で用いるには向かない．

2.2 Permutohedral Lattice を用いた高次元ガウシアンフィルタの高速計算法[4]

Andrew ら[4]は，Bilateral Grid のような手順が 3 つに分かれた手法を用いた Bilateral Filter に代表される高次元ガウシアンを用いた画像フィルタを高速に計算する方法を提案している．

Andrew らは，手順を Splat，Blur，Slice の 3 つに分け高速化を実現している．

2.2.1 Splat

Splat は，TG のグリッド作成のように，対象画像を別の空間で表現し直す処理である．しかし，TG のグリッド作成とは違い，図 5 の ように TG のグリッド作成で作成したデータを全ての要素が 1 のベクトル$\vec{1}$の直交補空間に射影したデータ構造を作成する．$\vec{1}$の直交補空間に射影することで空間の次元が 1 つ落ちるため，データのスパース性を解消することができる．このとき，データは図 5 に示すような単位単体を用いたデータ構造で表され，単体内部に画素が射影される場合，重心補間と呼ばれる補間方法を用いて単位単体の各頂点にデータが分散される．

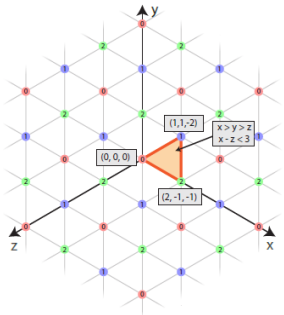


図 5 Permutohedral Lattice を用いたデータ構造(3 次元ガウシアンフィルタの例)(文献[4]より引用)

2.2.2 Blur

Blur は[1,2,1]の平滑化カーネルを用いて Splat で作成したデータ構造を単位単体の各辺に沿う方向に向かって平滑化する処理である．図 5 のデータ構造に平滑化を行う場合，x, y, zの 3 方向から[1,2,1]のカーネルを用いて平滑化を行う．

2.2.3 Slice

Slice は Blur を行った後のデータ構造から結果画像を復元する処理である．Splat で重心補間を用いてデータを分散させているため，重心補間の重みを用いた逆演算によって TU の分母と分子を復元し，割り算を行うことで結果画像を復元する．

3. 提案手法

LTU の計算が遅い原因としては以下の 2 つが挙げられる．

(1)shift-variant な画像フィルタ

(2)LMI の計算

提案手法では，(1)(2)を別の問題として考えそれぞれの解決手法を組み合わせることで LTU を高速化する方法を示す．

Vol.2020-CVIM-220 No.18
2020/1/24

3.1 高次元ガウシアンフィルタの高速計算法

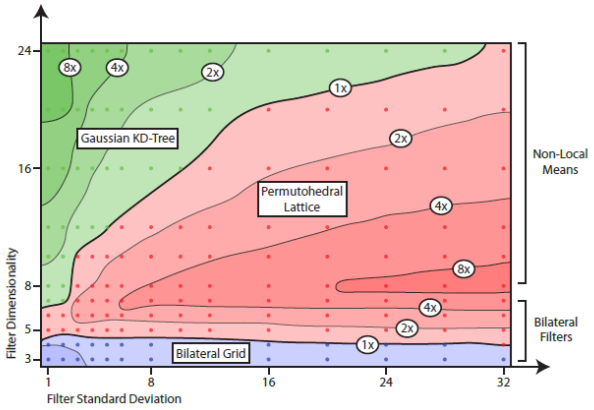


図 6 高次元ガウシアンフィルタの次元数ごとの有効手法比較結果(文献[4]より引用)

文献[4]によると，図 6 に示すように標準偏差σとガウシアンフィルタの次元数により有効な手法が異なることが示されている．

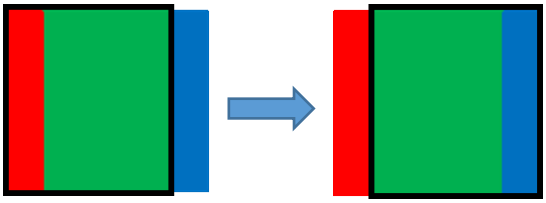
本研究で対象とする LTU は，フィルタの次元が，画像の空間x,yと可視光画像の輝度値r,g,b温度画像の輝度値iの 6 次元であるためσによらず Permutohedral Lattice を用いた高次元ガウシアンフィルタの高速計算法が有効である．そのため本研究では，Permutohedral Lattice を用いた高次元ガウシアンフィルタの高速計算法を用いる．

文献[4]では，Blur で用いる平滑化カーネルは[1,2,1]とされていたが，LTU にこれを用いると正常な計算結果を得られない場合が確認できたため，本研究では，幅 3 のガウシアンカーネルを用いることとする．

3.2 LMI の高速計算法

LMI の計算は，画像を小領域Sで区切りSを 1 画素ずつ移動させながら，Sの内部で相互情報量を計算するというものである．

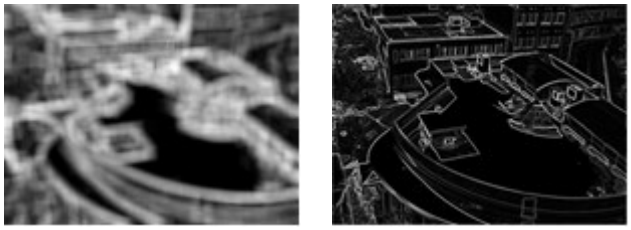
相互情報量を計算する際に求めるヒストグラムは，Sを隣に 1 画素移動させたときのものと大部分が共通している．そのため，隣のヒストグラムを計算する際には現在のヒストグラムから共通しない部分を除き，新しく追加される部分の画素を参照しヒストグラムに追加することで大幅にメモリ参照回数と演算回数を減らすことが出来る．図 7 の場合，まず点(x,y)を中心とするヒストグラムh_1を計算する．その後，点$(x+1,y)$のヒストグラムh_2を計算するとき，h_1とh_2は，緑の部分の輝度値が共通している．そのため，h_1から赤の部分のヒストグラムを除き，青の部分のヒストグラムを追加することでh_2を求めることが出来る．



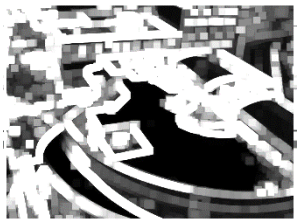
点(x,y)を中心とする小領域のヒストグラム計算　　点$(x+1,y)$を中心とする小領域のヒストグラム計算

図 7 隣接ヒストグラムの計算例

3.3 エッジ情報を用いたパラメータの動的変更法



a) LMI の画像化例　　b) エッジ情報の画像化例



c) エッジ情報の画像化例(膨張処理後)

図 8 LMI とエッジ情報の比較図

LMI を可視化したものとガイド画像に3×3のソーベルフィルタをかけたガイド画像のエッジ情報を比較すると図 8 の a)と b)のようによく似ていることがわかる．しかし，エッジの太さが違うため，膨張処理によって近い太さになるまでエッジを膨張させると c)になる．このエッジ情報を膨張させた c)を用いる．このとき，膨張処理には文献[5]で提案されているグレースケール画像用の膨張処理を用いる．

膨張処理後のエッジを用いた LTU の各重みは式(8)～(11)で計算される．このとき，膨張処理をした各チャネルのエッジはそれぞれ，e_R，e_G，e_B，e_Tとする．

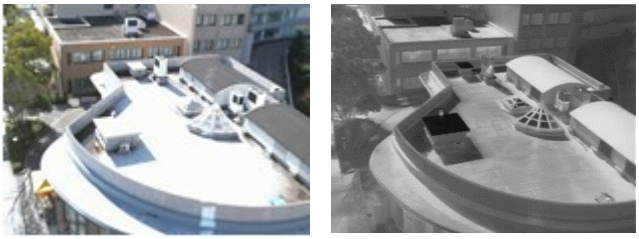
$$\sigma_R = \sigma\sqrt{\frac{(1+e_R\times e_T)}{e_R}} \tag{8}$$

$$\sigma_G = \sigma\sqrt{\frac{(1+e_G\times e_T)}{e_G}} \tag{9}$$

$$\sigma_B = \sigma\sqrt{\frac{(1+e_B\times e_T)}{e_B}} \tag{10}$$

$$\sigma_T = \sqrt{\frac{1}{\frac{1}{\sigma\sqrt{1+e_R\times e_T}}+\frac{1}{\sigma\sqrt{1+e_G\times e_T}}+\frac{1}{\sigma\sqrt{1+e_B\times e_T}}}} \tag{11}$$

4. 実験



a) ガイド画像(1151 × 863)　b) 対象画像(640 × 480)

図 9 実験に用いる画像対

表 1 実験に用いるパラメータ

	LTU(CPU)	LTU_LMI(GPU)	LTU_Edge(GPU
σ s	25	25	25
σ	10	10	2
σ e			5

表 2 実験環境

CPU	Intel® Core i7-6700K @4.00GHz × 8
GPU	nvidia GeForce GTX 980 Ti

提案手法を用いた場合，どの程度 LTU を高速化することが出来，どの程度の性能を発揮するかを確認するため，図 9 に示す画像対を用いて LTU を行った場合の実行時間と実行結果画像の品質を比較する．結果画像の品質比較には PSNR(Peak Signal to Noise Ratio)と水平微分ログヒストグラム，垂直微分ログヒストグラムを用いる．また，実験に用いるパラメータを表 1 に実験に用いた環境は表 2 に示す．

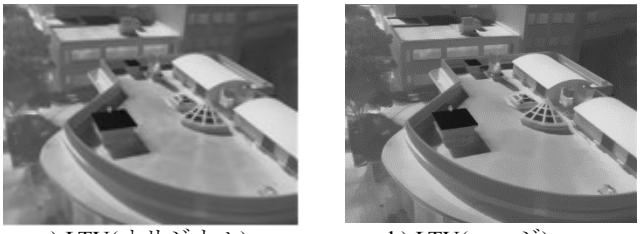
4.1 実行時間比較

表 3 実行時間の比較結果

	CPU	LMI	edge
実行時間[ms]	305887	1407	40

表 3 は，LTU を CPU 実装，GPU 実装，エッジ情報を用いた GPU 実装の 3 種類の実装方法で実行時間の比較をしたものである．CPU では約 5 分かかっていた処理が 3.1 節と 3.2 節の高速化法を組み合わせたものでは，約 217 倍速の 1407[ms]で，エッジ情報を使用し 3.1 節と 3.3 節の高速化法を組み合わせたものでは，約 7647 倍速の 40[ms]で実行できることが確認できる．

4.2 結果画像比較



a) LTU(オリジナル)　b) LTU(エッジ)

図 10 局所相互情報量を用いた場合とエッジ情報を用いた場合の LTU の出力画像比較

図 10 は表 1 のパラメータで LTU を計算したときの結果画像である．この画像対の PSNR を計算した結果，約 39[dB]である．PSNR は，30[dB]以上の値であるとお互いの画像はよく似ていると言えるという指標である．したがって，エッジ情報を用いて計算した LTU の結果画像は元の LTU の結果画像を高品質に再現できているといえる．

4.2.1 水平微分ログヒストグラムによる比較

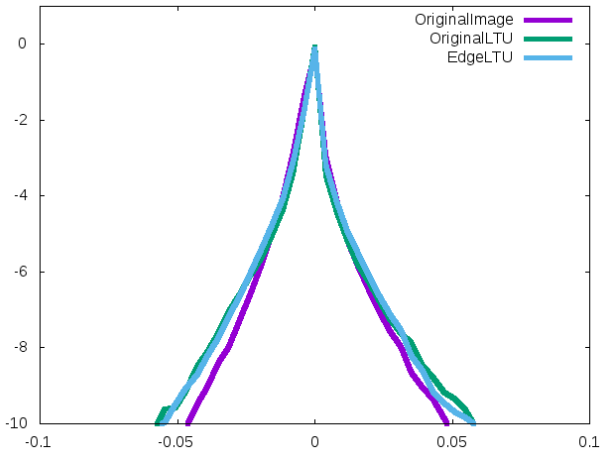


図 11 対象画像，通常の LTU の結果画像，エッジ情報を用いた LTU の結果画像の水平微分ログヒストグラム

水平微分ログヒストグラムは，画像を水平方向に微分し，対数をとったもののヒストグラムである．これは，横軸が 0 の付近が高く幅が狭い状態であるとノイズ除去性能が高いことを示し，裾の緒が広いと画像のテクスチャを残すことが出来ていることを示すと言われている．

図 11 は紫で描かれている元画像の線よりも緑で描かれている通常の LTU や水色で描かれているエッジ情報を用いた LTU のほうが横軸 0 付近が高く横幅が狭くなっているため，元画像よりもノイズが除去できているといえる．さらに裾の緒も横に長く広がっているため通常の LTU とエッジ情報を用いた LTU は共にテクスチャを残すことが出来ていると言える．また，通常の LTU とエッジ情報を用いた LTU の水平微分ログヒストグラムにほとんど差が見

られないため通常の LTU とエッジ情報を用いた LTU はほぼ同じ性能であると言える．

4.2.2 垂直微分ログヒストグラムによる比較

垂直微分ログヒストグラムは，水平微分ログヒストグラムの微分方向を水平方向から垂直方向に変えたもので評価方法は水平微分ログヒストグラムと同じである．

図 12 は紫で描かれている元画像の線よりも緑で描かれている通常の LTU や水色で描かれているエッジ情報を用いた LTU のほうが横軸 0 付近が高く横幅が狭くなっているため，元画像よりもノイズが除去できているといえる．さらに裾の緒も横に長く広がっているため通常の LTU とエッジ情報を用いた LTU は共にテクスチャを残すことが出来ていると言える．しかし，通常の LTU とエッジ情報を用いた LTU の裾の緒の広さに差があり，垂直微分ログヒストグラムで画像を評価した場合エッジ情報を用いた LTU は通常の LTU と同じ程度ノイズを除去できるが通常の LTU ほどテクスチャを保持することは出来ないと言える．

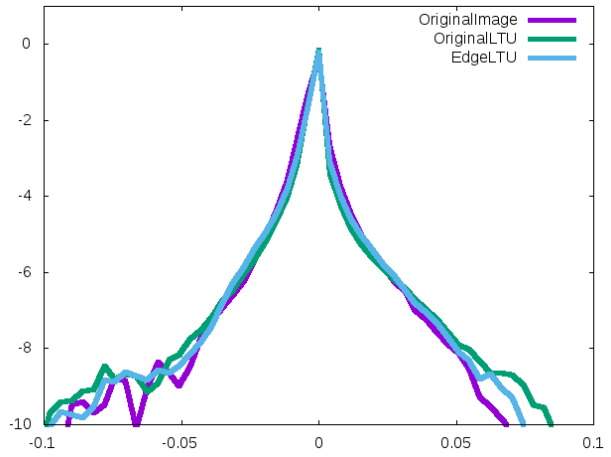


図 12 対象画像，通常の LTU の結果画像，エッジ情報を用いた LTU の結果画像の垂直微分ログヒストグラム

5. おわりに

本稿で提案した手法では，LTU の計算を LMI とよく似た結果を得ることが出来るエッジ情報を用いて GPU 並列実装することで，ノイズ除去性能をそれほど下げることなく大幅に実行時間を減らすことに成功した．しかし，エッジ情報を用いた LTU は通常の LTU ほどテクスチャを保持できない．また自動運転などの用途を想定するとまだ実行速度が遅いといえる．

今後は，図 13 に示すようにエッジ情報を用いた高速化法で時間がかかっているエッジの膨張処理，Splat，Blur の計算方法をより高速にすることで自動運転などの用途に利用可能な高速化方法を検討する．

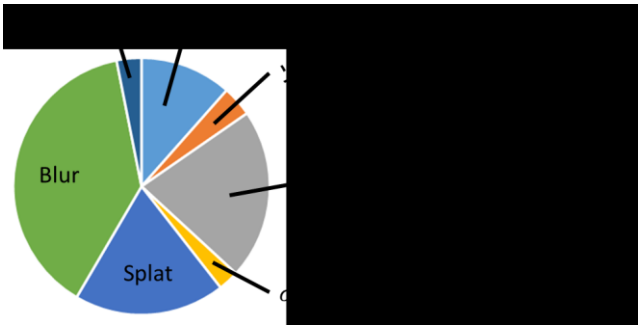


図 13 エッジ情報を用いた LTU の処理ごとにかかる計算時間の割合

参考文献

[1] Aleksandar S. Cvetkovic，Toshikazu Wada．“Local trilateral upsampling for thermal image,” ICASSP 2017，1577-1581.
[2] 門脇樊，和田俊和．“GPU を用いた Trilateral Upsampling の並 列実装” 研究報告コンピュータビジョンとイメージメディア (CVIM)，2018(6)，1-8.
[3] S. Paris，F. Durand．“A Fast Approximation of the Bilateral Filter using Signal Processing Approach,” International Journal of Computer Vision，vol. 81． no. 1， pp. 24-52， 2009.
[4] Andrew Adams，Jongmin Baek．Myers Abraham Davis，“Fast High-Dimensional Filtering Using the Permutohedral Lattice” Computer Graphics Forum 29，2010(2)，753-762.
[5] Octavia I． Camps，Tapas Kanungo，Robert M． Haralick，“Gray-Scale Structuring Element Decomposition”，IEEE TRASACTIONS ON IMAGE PROCESSING，Vol 5，No1，p111-120，1996.
[6] J. Sun，N.-N. Zheng，H. Tao，H.-Y. Shum，“Image hallucination with primal sketch priors,” IEEE Computer Society Conference on Computer Vision and Pattern Recognition，2003． Proceedings.， no. 2，pp. II-729，2003.
[7] C. Dong，C. Loy，K. He，X. Tang，“Learning a deep convolutional network for image super-resolution,”European Conference on Computer Vision，pp. 184-199，2014.
[8] S. C. Park，M. K. Park，M. G. Kang，“Super-resolution image reconstruction: a technical overview,”IEEE signal processing magazine，vol. 3，no. 20，pp. 21-36，2003.
[9] J. Kopf，M. F. Cohen，D. Lischinski，M. Uyttendaele，“Joint bilateral upsampling,”ACM Transactions on Graphics (ToG)，vol. 3，no. 26，p. 96，2007.
[10] J. Chen，S. Paris，F. Durand，“Real-time Edge-Aware Image Processing with the Bilateral Grid,”ACM SIGGRAPH conference proceedings，2007.