簡単なエッジ検出オペレータによる高速なエッジ検出アルゴリズム

5L−2

堀内 芳雄

日本アイ・ビー・エム株式会社

1. はじめに

エッジや線などの画像中の濃淡が急に変化している個所は、物体のなんらかの構造を反映していることが多く、重要な意味を持つ。たとえば、フォトレタッチソフトウェア「デジカメの達人」[1]が提供する「簡単選択ツール」は、物体の輪郭を大まかに指定するだけで適切な選択領域を作成する選択ツールであるが、選択領域の境界はIntelligent Scissors[2]と呼ばれる、エッジの強度と方向の情報を利用して画像中の境界線を検出するアルゴリズムによって作成されている。

エッジ検出にはいくつかの方法があるが、代表的なものに濃淡変化を積和演算によって求める差分型エッジ検出オペレータとテンプレート型エッジ検出オペレータ[3]とがある。いずれも、一般的なぼかしフィルタに比べれば小さなフィルタではあるが、エッジ検出の方向に応じて複数のフィルタ処理が必要となるため、処理に必要な計算コストは大きなものとなる。

本論文では、処理に要する計算コストが小さい簡単なエッジ検出オペレータを定義してあらかじめ出力を求め、その結果を組み合わせて、より複雑なエッジ検出オペレータを高速に処理するアルゴリズムについて述べる。このアルゴリズムによれば、Robinsonのテンプレート型エッジ検出オペレータの場合、8方向のテンプレートの処理に必要な算術演算命令を、積和演算を単純に実装した場合（以下、単純な処理方法と呼ぶ）に比べて約半分に減らすことができる。また、評価実験を行い処理時間を計測した結果、処理速度が約25%改善されることが確認された。

2. エッジ検出オペレータ

エッジは画像中の濃淡が急激に変化している個所にあるので、その濃淡変化を検出すればよい。もっとも素直な検出方法は、濃淡に関する微分を求めることである。デジタル画像の微分は差分で代用できるので、画素$f(i,j)$におけるxおよびy方向の微分成分、$f_x(i,j)$と$f_y(i,j)$、は以下のように積和演算で記述できる。

$$f_x(i,j) = \sum_k \sum_l f(i+k, j+l)\, h_x(k,l) \quad \text{...式1a}$$

$$f_y(i,j) = \sum_k \sum_l f(i+k, j+l)\, h_y(k,l) \quad \text{...式1b}$$

ここでh_x、h_yを差分型エッジ検出オペレータと呼ぶ。差分をとる画素を増やしてある程度平均効果を持つオペレータとして、以下に示すSobelのエッジ検出オペレータがよく知られている。

$$h_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \quad h_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

Fig 1, Sobelのエッジ検出オペレータ

この他に、理想的なエッジパターンを想定し、それと画像とのテンプレートマッチングを行うことでエッジを検出する、テンプレート型エッジ検出オペレータがある。想定するエッジパターンの違いによってさまざまなものが考えられるが、代表的なものとして、次に示すRobinsonのテンプレート型エッジ検出オペレータとPrewittのテンプレート型エッジ検出オペレータがある。

$$h_N = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \quad h_{NW} = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{pmatrix} \quad h_W = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \quad h_{SW} = \begin{pmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{pmatrix}$$

$$h_S = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \quad h_{SE} = \begin{pmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix} \quad h_E = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \quad h_{NE} = \begin{pmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{pmatrix}$$

Fig 2, Robinsonのテンプレート型エッジ検出オペレータ

$$h_N = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix} \quad h_{NW} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{pmatrix} \quad h_W = \begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix} \quad h_{SW} = \begin{pmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{pmatrix}$$

$$h_S = \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} \quad h_{SE} = \begin{pmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \quad h_E = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \quad h_{NE} = \begin{pmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{pmatrix}$$

Fig 3, Prewittのテンプレート型エッジ検出オペレータ

ここで、N、NW、W、SW、S、SE、E、NE、は方向を示す。

これらのテンプレート型エッジ検出オペレータはエッジの方向別（この例では8方向）にテンプレートが用意されており、式1と同様に画像との積和演算を行う。そして最大出力が得られたテンプレートの方向をエッジの濃度が変化する方向とし、その出力値の大きさをエッジ強度とする。

3. エッジ検出処理

Robinsonのテンプレート型エッジオペレータを例にとり、単純な処理方法と、本論文で提案する高速なアルゴリズムについて必要となる演算命令の数について比較する。

3.1 単純な処理方法

いま、ある画素とその周囲の8個の画素を以下のようにアルファベットa,b,c,...,iとして表す。

a	b	c
d	e	f
g	h	i

Fig 4, 画素とその近傍

これらの画素に対する8方向のテンプレートの出力値は、式1と同様の積和演算で求めることができ、単純な処理方法は以下の式のようになる。

$$f_N = a + 2b + c - g - 2h - i$$

$$f_{NW} = 2a + b + d - f - h - 2i$$

Fast Edge Detection using Simple Edge Operators

Yoshio Horiuchi

IBM Japan, Ltd., 1623-14 Shimotsuruma, Yamato-shi, Kanagawa-ken 242-8502, Japan

$$f_W = a \quad -c+2d \quad -2f+g \quad -i$$
$$f_{SW} = \quad -b-2c+d \quad -f+2g+h$$
$$f_S = -f_N$$
$$f_{SE} = -f_{NW}$$
$$f_E = -f_W$$
$$f_{NE} = -f_{SW}$$

式から明らかなように、Robinson のテンプレート型エッジ検出オペレータの場合には、最初の4つのテンプレート、h_N、h_{NW}、h_W、h_{SW}に対する出力結果を求めれば、残り4つの結果は符号を反転させることで得られる。

これらのフィルタ処理に必要な演算命令はコンパイラの最適化能力などにもよるが、加減算命令が12個、乗算命令が8個(定数のためシフト演算で代用可能)、符号反転命令4個、の、計36個である。

3.2 高速なアルゴリズム

以下のように、簡単な4つのエッジ検出オペレータを定義する。

$$h_1 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix} \quad h_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{pmatrix} \quad h_3 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{pmatrix} \quad h_4 = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Fig 5, 簡単なエッジ検出オペレータ

これら4つのエッジ検出オペレータとRobinsonのテンプレート型エッジ検出オペレータとの間には以下の関係式が成り立つ。

$$h_5 = h_1 + h_2 + h_3$$
$$h_6 = h_3 + h_4 - h_1$$
$$h_N = h_5 - h_6$$
$$h_{NW} = h_5 + h_2$$
$$h_W = h_5 + h_6$$
$$h_{SW} = h_6 + h_4$$
$$h_S = -h_N$$
$$h_{SE} = -h_{NW}$$
$$h_E = -h_W$$
$$h_{NE} = -h_{SW}$$

つまり、エッジ検出オペレータh_1、h_2、h_3、h_4、に対する出力値をあらかじめ求めておけば、その出力値を単純に加減算することによって、Robinson のすべてのテンプレートの出力値を得ることができる。

エッジ検出オペレータh_1、h_2、h_3、h_4の処理にはそれぞれ加算と減算1個ずつ、計8個の演算を要する。また、上述の関係式には加減算が8個、符号反転が4個あるので、すべてのテンプレートを処理するために必要な命令は20個となる。 3.1で述べた単純な処理方法に要する36個に比べて大幅に削減されていることが分かる。最初の4つのテンプレートh_N、h_{NW}、h_W、h_{SW}のみに着目すれば、一般の処理方法の半分の算術演算命令で処理が行える。

4. 評価実験

3章で述べた単純な処理方法と、本論文の高速なアルゴリズムのパフォーマンスを比較するために、実際の画像に対する処理実験を行った。使用した画像は64x64ピクセルの大きさで、500回のエッジ検出処理にかかる時間を計測した。(処理時間は5回の試行の平均値)

Table 1, 処理時間の比較

エッジ検出オペレータのタイプ	単純な処理方法(msec)	本論文の方法(msec)	改善率
Robinson Template	240	180	25.0%
Prewitt Template	220	180	18.2%
Sobel	190	140	26.3%

ただしテスト環境はIBM ThinkPad 570E、192MBメモリ、Windows 2000で、また、使用したコンパイラはMicrosoft Visual C++ 6.0である。

実験結果から、Robinsonのテンプレート型エッジ検出オペレータの場合、本論文の高速なエッジ検出アルゴリズムが単純な処理方法に比べて処理速度を25%向上させることが分かった。3.2で、必要となる算術演算命令数は約半分に削減される、と述べたが、実際の処理にはロードやストアなどフィルタの算術演算命令以外の命令が必要であり、全体では25%程度の改善にとどまったものと考えられる。

また、Prewittのテンプレート型エッジ検出オペレータ、および、Sobelのエッジ検出オペレータについて、3.2と同様の高速なアルゴリズムを導き出して、処理時間を比較する実験を行った。Prewittのテンプレート型エッジ検出オペレータで18.2%、Sobelのエッジ検出オペレータで26.3%、パフォーマンスが改善される、という結果が得られた。簡単なエッジ検出オペレータとの間の関係式は異なるが、本論文の高速なアルゴリズムの基本的な考え方である、簡単なエッジ検出オペレータを利用して複雑なエッジ検出オペレータを解く方法が、さまざま形のエッジ検出オペレータに広く適用可能であることを示すものと考えられる。

5. まとめ

本論文で述べたエッジ検出の高速なアルゴリズムは、単純な処理方法との算術演算命令の数の比較、および、評価実験の結果から、よく知られているエッジ検出オペレータのパフォーマンス改善に有効であることが確認された。

このアルゴリズムを任意のエッジ検出オペレータに適用する場合には、そのエッジ検出オペレータにあわせて、簡単なエッジ検出オペレータを定義してそれとの関係式を導き出し、処理を最適化することが必要となる。関係式の導出はある程度自動化されることが望ましく、今後、この手法について考察したい。

参考文献

[1] 「デジカメの達人」ウェブページ
http://www.ibm.co.jp/software/internet/dcmaster

[2] Intelligent Scissors for Image Composition
Eric N. Mortensen and William A. Barrett,
SIGGRAPH'95, Los Angeles, CA, Aug. 1995

[3] 画像処理標準テキストブック
(財)画像情報教育振興協会発行、
ISBN4-906665-10-1