4L-04

時系列ステレオ画像を利用した道路領域抽出とその高速化

原 智章† 中野 勝之‡ 奥富 正敏‡

†東京工業大学制御システム工学科 ‡東京工業大学大学院情報理工学研究科

1 はじめに

本研究室では自律走行車の実現に必要不可欠な道路領域認識を時系列ステレオ画像間の射影変換行列を利用して行っている [1]。射影変換行列を各時刻において更新し、前時刻の抽出結果を利用しているので、画像毎に単独で求めるよりもロバストな手法になっている。本論文では特に実時間での処理を目指したマルチスレッドなどによる処理の高速化に関して発表する。また、膨張 (収縮) 処理の効率的な手法についても提案する。

2 道路領域抽出法概要

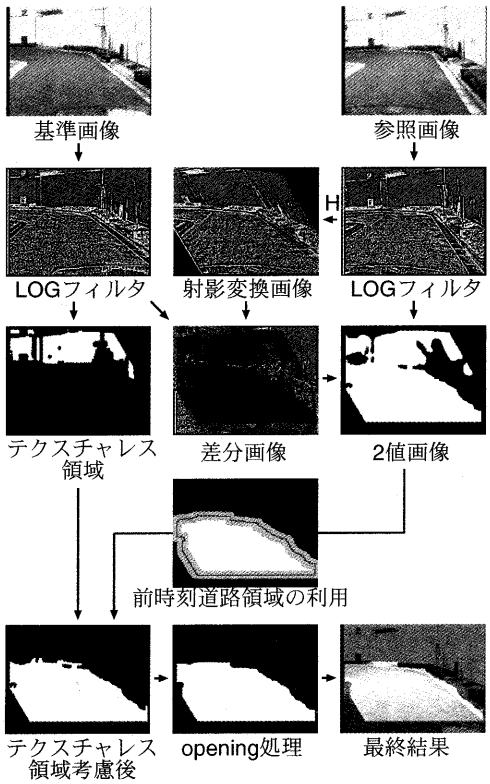

図 1: 処理の流れ

まず、入力画像にLOGフィルタをかけ、異なるカメラから得られた画像間の明るさの補正を行う。次に、対象とする道路平面についての参照画像から基準画像への射影変換行列 H を求める。参照画像を H によって変換し、基準画像とのマッチングをとる。サンプリング間隔の十分小さい時系列画像毎に処理を行っているので、前時刻道路領域を考慮することでテクスチャレスな領域に対してもロバストに推定が行える。最後に走行不可能な細線の領域を除去するために opening(収縮膨張) 処理を施す。

また、本手法では時系列画像間の射影変換行列も利用して処理をよりロバストなものにしている。

3 処理速度の向上

処理速度の向上のため、以下のような方法を用いた。

1. プログラムのマルチスレッド化
   複数の CPU を用いた並列処理。今回の実験では 2 つの CPU を使用した。
2. アルゴリズムの改良
   計算方法の改善、プログラム合理化
3. MMX 命令の利用 [2]
   intel CPU などに搭載された命令群で、一度の命令で同時に複数の変数を扱うことができる (SIMD、Single Instruction / Multipule Data)。本研究ではLOG フィルタで利用していて、4 つの変数を一度に扱っている。

3.1 でプログラムのマルチスレッド化、3.2 でアルゴリズムの改良例として、膨張 (収縮) 処理について述べる。

3.1 プログラムのマルチスレッド化

プログラムのマルチスレッド化の方法は、以下の 2 つのタイプに分けられる。

1. 独立な異なる関数を同時に処理する
2. 画像を複数に分け、それぞれを同時に処理する

筆者らはすでに画像処理のマルチスレッド化を容易に実現できるプログラミング環境を提案している [3]。これを利用すれば、前者のマルチスレッド化は容易に実現できる。一方、後者の場合、特に分割した画像の境界部分において、メモリへのアクセス、書き込みが同時に行われるといった干渉が起きないように工夫する必要がある[1] 。

また、例えば領域のラベリング (図 2(a)) では、一つの領域が境界面にまたがって存在するときは 2 つの別

Extraction of Road Region Using Sequential Stereo Images and Acceleration of the Computing Speed.
Tomoaki HARA†,Katsuyuki NAKANO‡, Masatoshi OKUTOMI‡. †Department of Control and Systems Engineering, Tokyo Institute of Technology. ‡Graduate School of Information Science and Engineering, Tokyo Institute of Technology.

[1] アクセス、書き込みの順序によって結果が変わってしまう

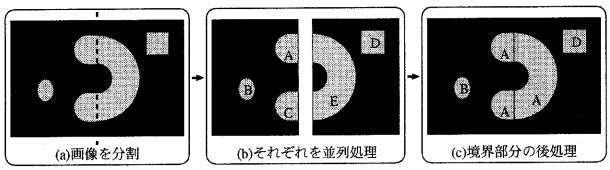

図 2: 領域のラベリングにおけるマルチスレッド化例

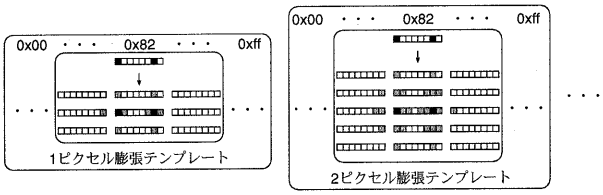

図 3: 膨張テンプレート (4 近傍の例)

の領域となってしまう (図 2(b))。この場合は後で隣り合う領域をまとめる処理を行うことで結果の整合性をとる。(図 2(c))。

3.2 膨張 (収縮) 処理における高速化の提案

従来の n 画素膨張処理では (収縮処理も同じ原理)、画像を走査して膨張したい画素を見つけたらその周りを同じ色にするということを 1 画素ずつ n 回繰り返していた。

ここでは、一度に 8 画素処理し、かつ画像走査回数の少ない手法を提案する。以下の手順で処理を行う。

0. 準備：膨張テンプレート作成 (図 3)
   膨張テンプレートとは、1byte で表される 1bit×8 の 8 つの画素のパターン (0x00〜0xff) によって、膨張処理を行うとその周りの画素がどう変化するかを記述したものである。膨張する画素数 n に対応したテンプレートをあらかじめ用意しておく。
1. 画像の圧縮 (pgm → pbm)
   (1×8) の 8 つの画素を 1byte で表すように変換
2. テンプレートによる膨張処理
   圧縮した画像を 1byte ずつ取り出し、その値に対応したテンプレートパターンを出力画像に埋め込む。
3. 画像の解凍 (pbm → pgm)

表 1 に従来手法、提案手法、提案手法をマルチスレッド化したときの処理時間を示す。膨張テンプレートは $n = 1 \sim 8$ を用意し、10,20 画素膨張ではこれらのテンプレートを組み合わせて処理した。従来手法に比べ、提案手法では処理時間が大幅に減少した。一度に 8 画素処理し、また、テンプレートを用いることで画像を走査する回数が減ったためである。また、マルチスレッド化によってさらに約半分になっていることがわかる。

4 処理結果

以上ような高速化を行って道路領域抽出処理を行った。PentiumII450MHz×2, Memory 512MB,OS は Linux

表 1: 膨張処理でかかる時間の違い (msec)

膨張 pixel 数	1	5	10	20
従来手法	19.320	94.480	195.017	414.066
提案手法	1.250	4.658	8.736	16.802
multi thread	0.835	2.482	4.468	8.645

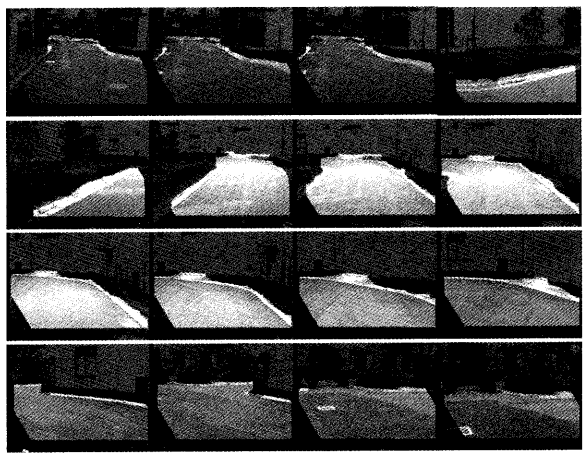

図 4: 道路領域抽出結果

を用いた。以前では平均 9.12[sec/frame] かかっていたのが MMX 命令の利用、アルゴリズムの改良により平均 3.78[sec/frame] になった。さらにマルチスレッド化して平均 2[sec/frame] に向上した。

図 4 に道路領域抽出結果を左から右へ順に示す。また、本研究室のホームページ[2] で処理結果を動画で見ることができる。

5 おわりに

時系列ステレオ画像を利用した道路領域抽出の処理速度向上を行った。また、膨張収縮処理の効率的な手法を提案した。この道路領域抽出法を応用すれば任意の平面を追跡でき、ヘリコプタの離着陸の自動化など、様々な分野で利用できると思われる。処理速度の大幅な改善を遂げたが、実時間での処理にはさらなる改善が必要である。

参考文献

[1] 奥富正敏, 丸山純一, 中野勝之. ステレオ動画像を用いた視覚誘導のための平坦部の連続推定. 情報処理学会研究報告 CVIM, Vol. 2000, No. 82, pp. 17–24, 2000.

[2] Vladimir Kravtchenko. Using mmx technology in digital image processing. Technical Report and Coding Examples.
http://www.cs.ubc.ca/spider/vk/mmx.html.

[3] 中野勝之, 奥富正敏. マルチスレッドを利用したビジュアルナビゲーション研究開発環境. 第 5 回ロボティクスシンポジア, pp. 433–438. RSJ, SICE, JSME, 3 月 2000.

[2] http://www.ok.mei.titech.ac.jp/res/res-j.html♯VN