キャッシュヒット率を考慮したステレオマッチングの並列処理

非会員 中 澤 健 亮 (千葉工業大学)

非会員 古 屋 穂 高 (千葉工業大学)

正 員 前 川 仁 孝 (千葉工業大学)

正 員 伊與田 光 宏 (千葉工業大学)

Parallel Processing Scheme of Stereo Matching Considering Cache Hit Rate

Nakazawa Kensuke, Non-member, Furuya Hodaka, Non-member, Maekawa Yoshitaka, Member, Iyoda Mitsuhiro, Member (Chiba Institute of Technology)

Stereo matching takes a long computation time. It is expected that we can speed up stereo matching by parallel processing. One of the parallel processing methods is line partitioned which is dividing the image domain into one line. This method assigns each line to a processor when searching point. Then, the load of the matching processing is distributed almost equally. However, it has the problem of cache memory in the case of read miss for dividing large domains. Therefore, this paper proposes the speed up scheme to improve the hit ratio of cache memory by dividing the domain into small blocks. Lastly, the effectiveness of the proposed method is shown by revealing the experimental results on the shared memory multiprocessor named 'Sun Enterprise 4500'.

キーワード：ステレオマッチング，並列処理，キャッシュメモリ，共有メモリ型マルチプロセッサ

Keywords: Stereo Matching, Parallel Processing, Cache Memory, Shared Memory Multiprocessor

1. ま え が き

ステレオマッチングは、複数の画像を用い三角測量の原理から3次元形状を求め、画面内すべての点の3次元情報を求める手法である^{(1)~(5)}。従来より用いられている手法に、2乗残差法⁽¹⁾⁽⁵⁾や正規化相互相関法⁽¹⁾⁽⁵⁾⁽⁶⁾などがある。2乗残差法は、基準画像と参照画像間でテンプレート内の2乗差を計算し対応点を探索する手法で、計算時間は早いですが、マッチングの信頼度は低いことが知られている⁽¹⁾⁽⁵⁾。一方、正規化相互相関法は、ステレオ画像間で同じ場所を示す部分の相関値が高くなることを利用して画像間の対応点を求める手法で、2乗残差法と比べマッチングの信頼度は高いが、3次元情報を求めるのに繰り返し計算を必要とするため処理時間が遅いことが知られている⁽¹⁾⁽⁵⁾⁽⁶⁾。そこで、マッチング処理の信頼度が高く処理時間がかかる正規化相互相関法を用いたステレオマッチングの高速化が必要とされている。

ステレオマッチングの高速化手法の1つに並列処理がある。共有メモリ型並列コンピュータを用いた並列処理手法に、処理画像をプロセッサ数で等分し、分割した画像ブロック領域に対しプロセッサを割り当て並列処理する手法⁽⁷⁾⁽⁸⁾⁽¹⁰⁾⁽¹⁸⁾⁽¹⁹⁾や、画像領域を1ラインずつに分割し、1

ラインごとに1プロセッサをサイクリックに割り当て並列処理する手法^{(8)~(10)}、画像領域を1ラインずつに分割し1ラインごとに1プロセッサをダイナミックに割り当て並列処理する手法⁽⁷⁾⁽⁸⁾などがある。処理画像をプロセッサ数に等分にブロック分割する手法は、分割された画像ブロック内の階調値差によりマッチング処理量が決まり処理時間に大きな偏りが生じる。ライン分割した画像領域をサイクリックに割り当てる手法とダイナミックに割り当てる手法は、マッチング処理の負荷は分散されるが、並列処理する際、基準画像の処理ラインが変わるとキャッシュメモリが参照できなくなる。また、処理する画像サイズが大きくなると画像データがキャッシュメモリに格納しきれず、探索した画像データがフラッシュされキャッシュメモリに格納されていた画像データが参照できなくなる。以上のことからマッチング処理の速度向上が図れないという問題がある。

そこで本論文では、キャッシュメモリの有効利用^{(9)~(12)}を行い、マッチング処理の負荷を均等にして効率的な並列処理を行うために、画像分割するとき、探索した画像データをキャッシュメモリに格納できるように画像領域を小ブロックに分割する手法を提案する。また、提案する手法を共有メモリ型並列コンピュータ Sun Enterprise 4500 上に実装し有効性を確認する。

2. 相関法を用いた対応点探索

図1 テンプレートマッチングの基準画像 $f(x, y)$
Fig. 1. Object Image of Template Matching

図2 テンプレートマッチングの参照画像 $g(x, y)$
Fig. 2. Search Image of Template Matching

テンプレートマッチングに基づくステレオマッチングの基準画像 $f(x, y)$ を図1, 参照画像 $g(x, y)$ を図2に示す。図1中の $(2K + 1) \times (2L + 1)$ はテンプレートサイズを表し, (x_L, y_L) はその中心画素とする。また, 図2中の $(M + 1) \times (2N + 1)$ はサーチエリアを表し, サーチエリア上に (x_R, y_R) を中心画素とするテンプレートを定める。

相関法を用いた対応点探索は, 図1の基準画像 $f(x, y)$ に対して, 図2の参照画像 $g(x, y)$ 中で最も相関係数 $T (0 \leq T \leq 1)$ が高くなる点を対応点とする手法である。具体的には, 基準画像 $f(x, y)$ 上の $(2K + 1) \times (2L + 1)$ 画素のテンプレートに対し, 参照画像 $g(x, y)$ 上で最も一致度の高い位置を求め, その位置でのテンプレートの中心画素 (x_R, y_R) を対応点とする。ただし, テンプレートマッチングを行う範囲は参照画像での座標 (x_L, y_L) を右端とする $(M + 1) \times (2N + 1)$ 画素のサーチエリア内を探索する。ここで, 上下方向に視差がない場合, 左右の画像で対応する

点は同一の高さの走査線上に存在するため $N = 0$ とすることもできる。しかし, 上下方向の若干のずれを考慮する場合は, N の値を変化させることで上下のずれについても対応点探索が可能になる。また, マッチングの際の評価値として, 相関係数の計算式である式(1)を正規化した式(2)を利用し画素ごとの相関係数を求める^{(1)(5)(6)(15)~(17)}。ただし, 参照画像 $g(x, y)$ のテンプレートの中心座標を (x, y) , 基準画像を f , 参照画像を g としている。

実際に対応点を探索するには, 参照画像上のテンプレートの中心 (x, y) を $(2K + 1) \times (2L + 1)$ の範囲を1画素ずつずらしながらラスタ走査し, 基準画像上のテンプレートの中心 (x, y) との相関係数 T が一番高くなる点を探すため, 式(2)を繰り返し計算する。

$$R_{fg}(x, y) = \sum_{i=-K}^K \sum_{j=-L}^L |f(x+i, y+j)g(x, y)| \cdots (1)$$

$$T = \frac{R_{fg}(x, y)}{\sqrt{\rho} \sqrt{\varrho}} \cdots (2)$$

ただし, ρ, ϱ は以下に示す通りである。

$$\rho = \sum_{i=-K}^K \sum_{j=-L}^L f(x+i, y+j)^2$$

$$\varrho = \sum_{i=-K}^K \sum_{j=-L}^L g(x, y)^2$$

3. 画像領域の分割手法

本章では従来より提案されている画像領域を分割する手法と分割した画像領域をプロセッサに割り当てる手法について述べる。

〈3・1〉ブロック割り当て 本節では, ブロック割り当て手法について述べる。図3はブロック割り当て手法⁽⁷⁾⁽⁸⁾⁽¹⁰⁾⁽¹⁸⁾⁽¹⁹⁾により, 仮にプロセッサを4台用いた場合の例

図3 基準画像のブロック割り当て例
Fig. 3. An Example of Block Assignment

を示す。基準画像をプロセッサ数で等分割し1ブロックに対して1プロセッサを割り当てマッチング処理する。

図3に示す様に、プロセッサ4台でマッチング処理する場合、仮にプロセッサ1のみがマッチング処理の重い画像領域を担当したとする。他の3つのプロセッサはプロセッサ1と比べて処理が軽いので処理時間がかからない。すると、プロセッサ1は他の3つのプロセッサが処理を終えてもプロセッサ1が担当するブロックはマッチング処理を続けなければならない。つまり、本手法では、ある領域にマッチング処理する画像が偏る場合、マッチング処理するときに画像領域がプロセッサ数で均等に割り当てられるので、あるプロセッサに処理が偏り計算の負荷が分散されず処理時間がかかることがある。

〈3・2〉サイクリック割り当て 本節では、画像領域をラインごとに分割しサイクリックに割り当てる手法^{(8)~(10)(13)}について述べる。図4は、仮にプロセッサを4台使用してサイクリックに割り当てた例を示す。基準画像領域は1ラインごとに分割され、プロセッサ0、プロセッサ1、プロセッサ2、プロセッサ3、プロセッサ0、プロセッサ1、...とプロセッサは順々に割り当てられ、各ラインごとにマッチング処理をする。

前節のブロック割り当て手法では、プロセッサ1台が処理する画像領域はブロックごとに分割されているので、マッチング処理に偏りが生じることがあるが、サイクリック割り当てでは、マッチング処理する領域に対して、1ラインごとにプロセッサを割り当てるため、画像中に処理の重い領域が存在しても4台のプロセッサに処理が分散する。しかし、画像領域を1ラインごととサイクリックに割り当てるため、実行前に処理するラインが決定するので、他のプロセッサの処理が終っても速く処理が終ったプロセッサが処理の遅いプロセッサに処理時間を合わせることで、全体としての処理が遅くなることもある。

Processor0
Processor1
Processor2
Processor3
Processor0
Processor1
Processor2
Processor3
Processor0
Processor1
Processor2

図4 基準画像のサイクリック割り当て例
Fig. 4. An Example of Cyclic Assignment

〈3・3〉ダイナミック割り当て 前節で述べたサイクリックに割り当てる手法の問題点を解決するためにダイナ

ミックに割り当てる手法⁽⁷⁾⁽⁸⁾が提案されている。図5は、画像領域をラインごとに分割しプロセッサをダイナミックに割り当てマッチング処理する例を示す。仮にプロセッサ4台でダイナミックに割り当てたとき、最初の4ラインは4台のプロセッサを順に割り当てて、その後のラインはプロセッサが処理を終えた順にラインの処理を割り当てる。サイクリックに割り当てたときは、速く処理が終ったプロセッサが、処理の遅いプロセッサに処理時間を合わせることで処理時間がかかることがあるが、ダイナミックに割り当てたときは、サイクリックに割り当てたときと比べて負荷が分散され処理が速くなる。

図5 基準画像のダイナミック割り当て例
Fig. 5. An Example of Dynamic Assignment

4. キャッシュを考慮した分割手法

ステレオマッチングの処理を並列処理すると考えたとき、画像領域を1ラインごとにサイクリック又はダイナミックに割り当てる手法では、ラインごとにマッチング処理するため、基準画像上の処理するラインが移動した場合、参照画像上のデータがキャッシュメモリに格納されていると思われるラインの画像データを参照できなくなり、処理が遅くなることもある。また、処理する画像サイズが大きくなると画像データがキャッシュメモリに格納しきれず、探索した画像データがフラッシュされキャッシュメモリに格納されていた画像データが参照できなくなる。図6は、ラインごとに並列処理した場合における参照画像上のサーチエリアとテンプレートの動きを示し、サーチエリア内をテンプレートが対応点を探索する様子を示す。また、実線で区切られているのは1プロセッサがマッチング処理する1ラインごとに分割したことを示す。図6に示すように、参照画像上のサーチエリアが割り当てられたラインを探索し次のラインに移動すると、現在キャッシュメモリ内に格納されている画像データは参照されことなく次のラインを処理することになり、速度向上の妨げになっていることが分かる。そこで、ライン分割にキャッシュヒット率をあげるデータローカライゼーション手法を適用すると、画像データを

図 6 ライン分割によるキャッシュの様子
Fig. 6. The Cache Memory Division by the Line

図 7 ブロック分割によるキャッシュの様子
Fig. 7. The Cache Memory Division by the Block

キャッシュメモリに格納できるように画像領域をライン分割し、分割した1ラインに対してキャッシュメモリサイズを考慮して分割する。しかし、ライン分割にデータローカライゼーション手法を用いても大部分の画像データが有効利用できず、キャッシュメモリの利用効率が悪くマッチング処理の高速化が図れない。

一方、ブロック分割手法は、図7に示すように、プロセッサ数で画像領域を均等に分割し、1プロセッサが1ブロックを割り当てる。図7は、ブロック分割した参照画像上で、サーチエリアとテンプレートを示し、サーチエリア内をテンプレートが対応点を探索する様子を示す。また、点線で区切られているのは画像領域を1ラインごとに分割したことを表し、実線で区切られているのは1プロセッサが処理する範囲を示している。図7から分かるように、割り当て

られたブロックをマッチング処理するサーチエリアが1ライン目の処理が終わって下のラインへ移動しても1ライン目で探索した参照画像上の画像データがキャッシュメモリに多く格納されている可能性があり、キャッシュメモリ内の画像データを有効利用できる。しかし、処理の単位が大きなブロックになると対象物体の位置により、マッチング処理量がプロセッサ間で大きく異なるので、負荷分散に問題が生じ、並列処理による高速化が図れない。また、仮にキャッシュメモリが16kbyte 使えると仮定すると、例えば、今回評価に用いた画像のように1ラインが614画素あるような画像をマッチング処理する場合、サーチエリアサイズ25×5とすると128画素目の処理を行った段階でキャッシュメモリに空き領域がなくなり参照されたデータがフラッシュされラインが移動しても参照されなくなり、高速化の妨げに

図 8 小ブロック分割によるキャッシュ参照
Fig. 8. Reference of the Cache Depended Division by the small Block

なる。

そこで本論文では、以上の問題点を解決するために小領域分割法を提案する。小領域分割法は、マッチング処理の負荷分散を行い、キャッシュメモリの有効利用をするように、画像分割をするとき、画像領域を横方向はプロセッサ数で均等に分割し、縦方向は横方向で分割された画像サイズに対してキャッシュメモリの容量を考慮して、小ブロックに分割する。このように、小ブロックに分割することで、参照画像上を探索した画像データを共有メモリにアクセスせず処理するので、データ転送オーバーヘッドが小さくなりキャッシュメモリを有効利用できる。以上のことから小領域分割法は、分割されたブロック内を1ラインずつマッチング処理する。ブロック内でプロセッサが次のラインを探索するとき、ブロック分割のときと同様にキャッシュメモリに必要なデータが残っている可能性が高いので高速化ができる。図8は、仮にプロセッサ4台でマッチング処理したときの様子を示す。横方向はプロセッサ4台で画像領域を均等に分割し（ブロック分割と同様に分割）、縦方向は横方向で分割された画像サイズに対してキャッシュメモリの容量を考慮して分割すると、図に示すような実線で区切られた小領域に分割したブロックになる。また、サーチエリア内をテンプレートが対応点を探る様子を示す。図8より1ライン目をマッチング処理し、そのときの画像データがキャッシュメモリに格納される。処理する領域が小さい分、ブロック内でラインが変わってもキャッシュメモリ内のデータは格納されているのでキャッシュの参照ができキャッシュメモリが有効利用され処理の高速化が図れる。

5. 実験結果

本論文では、提案する手法の有効性を評価するためにダイナミック割り当て手法と小領域分割法を比較し、実際に並列処理した結果について述べる。

〈5・1〉 実行環境 提案する手法の有効性を評価するために Sun Enterprise 4500 を用いた。本機は、CPU に L1 キャッシュ 16kbytes、リードキャッシュラインサイズ 16bytes を持つ 400MHz の UltraSPARC-IIs が 14 台接続された主記憶共有型のマルチプロセッサである。

〈5・2〉 処理時間測定 はじめに、表1は画像サイズ 614×410 の処理時間を示す。ここで、マッチング処理に用いたサーチエリアとテンプレートのサイズは、相関係数の平均を測定して求める。また、2章で述べたように上下方向の若干のずれを考慮して N の値を 2 とすることで、サーチエリアサイズは 25×5、テンプレートサイズは 5×3 と決定する^{(10)~(12)}。

評価の結果、小領域分割法によりプロセッサ1台でステレオマッチング処理をしたときの処理時間は 465.63[s]、プロセッサ14台で並列処理したときの処理時間は 33.37[s] となり、逐次処理と比べて 13.95 倍高速化された。また、プ

表 1 画像サイズ 614×410 の処理結果
Table 1. The processing result of 614×410

プロセッサ数	ダイナミック 割り当て [s]	小領域分割法 [s]
1	464.29	465.63
2	234.83	232.94
3	158.03	155.60
4	120.24	116.57
5	97.24	93.20
6	82.03	77.68
7	71.31	66.59
8	63.04	58.33
9	56.67	51.87
10	51.62	46.70
11	47.48	42.45
12	44.07	38.92
13	41.11	35.92
14	38.62	33.37

ロセッサ 14 台で並列処理した際の小領域分割法とライン分割のダイナミック割り当て手法を比較すると、小領域分割法を用いた場合、13.6%の速度の向上が確認できた。

また、プロセッサ 1 台のときのマッチング処理で、小領域分割法よりライン分割によるダイナミック割り当て手法の方が若干処理時間が速かった。これは、ダイナミックに割り当てることにより負荷が分散されるが、提案手法では、たとえ小さな領域でもブロック単位で割り当てているので、マッチング処理に偏りができ処理時間を若干要したからと思われる。

次に、表 2 は画像サイズ 768 × 512 の処理時間を示す。評価の結果、小領域分割法によりプロセッサ数 1 でステレオマッチング処理したときは 832.02[s]、プロセッサ 14 台で並列処理したとき処理時間は 59.79[s] となり、逐次処理に比べて 13.92 倍高速化された。また、プロセッサ 14 台のとき、小領域分割法とライン分割によるダイナミック割り当て手法を比較すると小領域分割法を用いた場合、9.6%の速度向上が確認できた。画像サイズ 768 × 512 においても同様にプロセッサ数 1 のときの小領域分割法がライン分割によるダイナミック割り当て手法より処理が遅くなったが、プロセッサ 2 台以上では速くなることが確認できた。

表 2 画像サイズ 768 × 512 の処理結果
Table 2. The processing result of 768 × 512

プロセッサ数	ダイナミック 割り当て [s]	小領域分割法 [s]
1	830.17	832.02
2	422.03	417.45
3	282.25	279.28
4	213.92	208.78
5	172.23	167.25
6	144.55	139.25
7	124.97	119.30
8	110.36	104.59
9	98.78	92.87
10	89.65	83.66
11	82.23	76.01
12	75.85	69.72
13	70.49	64.36
14	66.12	59.79

最後に、表 3 は画像サイズ 1075 × 717 の処理時間を示す。評価の結果、小領域分割法によりプロセッサ 1 台でステレオマッチング処理したときは 1158.69[s]、プロセッサ数 14 で並列処理したとき処理時間は 82.89[s] となり、逐次処理に比べて 13.98 倍高速化されることが確認できた。また、プロセッサ 14 台のとき、小領域分割法とライン分割によるダイナミック割り当て手法を比較すると小領域分割法を用いた場合、10.1%の速度向上が確認できた。画像サイズ 1075 × 717 においても同様に、プロセッサ 1 台のときの小領域分割法は、ダイナミック割り当て手法より処理が遅くなったが、プロセッサ 2 台以上で並列処理した場合は、小領域分割法の処理が高速化していることを確認できた。

表 3 画像サイズ 1075 × 717 の処理結果
Table 3. The processing result of 1075 × 717

プロセッサ数	ダイナミック 割り当て [s]	小領域分割法 [s]
1	1150.69	1158.69
2	581.94	579.88
3	391.96	386.60
4	296.26	290.16
5	238.86	231.55
6	200.86	193.46
7	173.86	165.66
8	153.38	145.03
9	137.52	128.85
10	124.77	115.93
11	114.37	105.48
12	105.66	96.63
13	98.34	89.20
14	92.18	82.89

以上の結果よりいずれの場合でも今回用いたサーチエリアサイズを使用したとき、小領域分割法を用いた場合ライン分割によるダイナミック割り当て手法より、並列処理する際のキャッシュメモリの参照が多くなったため処理の高速化を図ることが確認できた。また、今回用いたサーチエリアは画像サイズ 614 × 410 に適したサイズだったので、画像サイズ 768 × 512、1075 × 717 では、ライン分割のダイナミック割り当て手法と小領域分割法でプロセッサ 14 台のときの処理の向上率が画像サイズ 614 × 410 に比べ悪かった。本来、画像サイズ 768 × 512、1075 × 717 にも適したサーチエリアサイズがある^{(9)~(11)}。しかし、サーチエリアを大きくとることによって、キャッシュメモリを参照する領域が大きくなるので、キャッシュメモリを参照する前にフラッシュされてしまう可能性がある。そこで、キャッシュメモリの容量を計算に入れる必要があると思われる。

6. おわりに

本論文では、共有メモリ型並列コンピュータ上のステレオマッチングにキャッシュヒットを多くすることを考慮した領域分割法を提案した。提案した小領域分割によるステレオマッチングの手法を用いた場合、ライン分割のダイナミック割り当て手法と小領域分割法を比較した結果約 9.6%~13.92%の処理速度向上、小領域分割法でプロセッサ 1 台と 14 台のとき並列効果は 13.92 倍~13.98 倍の速度向上が得られた。これにより共有メモリ上でのステレオマッチングの高速化には、キャッシュヒットを考慮した小領域分割法の有効性を示した。

また、本論文において使用したパラメータは、テンプレートサイズが 5 × 3、サーチエリアサイズを 25 × 5 として測定を行ったが、最適なサイズは画像サイズによって違ってくる。そのため対応点となった画素からどのくらいの近傍に次の対応点があるかを予測してサーチエリアのサイズを変更する機能を開発することが必要である。

(平成 13 年 7 月 13 日受付, 同 14 年 2 月 4 日再受付)

文 献

- (1) 安居院猛, 長尾智晴:「画像の処理と認識」, 昭見堂, (1992)
- (2) 出口光一郎:「画像と空間」, 昭見堂, (1991)
- (3) 土屋豊, 深田陽司:「画像処理」, コロナ社, (1990)
- (4) 原島博, 中嶋正之, 山本正信:「グラフィックとビジョン」, オーム社, (1996)
- (5) 本田庸悟:「画像処理と視覚認識」, オーム社, (1995)
- (6) 林高志:「実環境 3 次元計測のためのステレオマッチング」, (2000)
- (7) 美濃導彦:「並列画像処理」, コロナ社, (1999)
- (8) 古屋穂高, 羽田泰啓, 宮本和則, 高田勝久, 前川仁孝, 伊與田光宏:「ワークステーションクラス上におけるステレオマッチングの並列処理」, 情報処理学会第 61 回全国論文集, Vol.2, pp.207-208, (2000)
- (9) 和泉秀幸, 佐々木和司, 水野政治, 中嶋克人:「SMP での SAR 画像再生処理の並列化〜キャッシュを活かしたコーナーターンの高速化〜」, 情報処理学会第 60 回全国論文集, Vol.1, pp.53-54, (2000)
- (10) 柿沼武史, 星仰:「サンプリング法によるステレオマッチングの並列処理」, 情報処理学会第 60 回全国論文集, Vol.2, pp.227-228, (2000)
- (11) 豊田誠, 星仰:「超高速パターンマッチングと応用のためのソフトウェア開発, 創造的ソフトウェア育成事業及びエレクトロニック・コマース推進事業」最終成果発表会論文集, pp.597-602, (1998)
- (12) Shigeaki Iwasa, et al: SSM-MP: "More Scalability in Shared-Memory Multi-Processor", Proc.ICCD, (1995)
- (13) 福地雄史, 石塚章子, 和泉秀幸:「マルチプロセッサ対応 UNIX 上での並列プログラム開発支援環境の開発」, 第 48 回情報処理学会全国大会 2G-9, (1994)
- (14) 星仰, 矢野正謹:「衛星画像の並列ステレオマッチング処理と標高誤差」, 情報処理学会第 60 回全国論文集, Vol.2, pp.225-226, (2000)
- (15) 日本リモートセンシング研究会:「画像の処理と解析」, 共立出版, (1981)
- (16) 星仰, 山下記久雄:「SPOT ステレオ画像によるオルソフォト画像処理」, 情報処理学会第 40 回全国大会講義論文集, (2000)
- (17) MARY P.ROTTINO: "Guide to Parallel Programming on Sequint Computer Systems", Prentice-Hall, (1992)
- (18) 伊藤千秋, 大澤範高, 弓場敏嗣:「画像処理利用ベンチマークによる分散記憶型並列計算機の評価」, 情報処理学会第 52 回全国論文集, Vol.6, pp.141-142, (1996)
- (19) 工業技術院監修:「SPIDER Users' Manual」, 情報技術コンソーシアム, (1982)

前川 仁 孝 (正員) 1967 年生。1990 年 早稲田大学理工学部電気工学科卒業。1992 年 同大学大学院理工学研究科電気工学専攻修士課程修了。1993 年 日本学術振興会特別研究員。1994 年 早稲田大学理工学部助手。1998 年 千葉工業大学着任。同大学情報科学部情報工学科助教授, 現在に至る。博士(工学)。各種アプリケーションの並列処理に関する研究に従事。情報処理学会会員。

伊與田 光 宏 (正員) 1955 年生。1978 年 慶應義塾大学工学科卒業。1983 年 同大学大学院工学研究科博士課程電気工学専攻修了。同年, 千葉工業大学着任。同大学情報科学部情報工学科教授, 現在に至る。工学博士。電子情報通信学会, 情報処理学会, レーザー学会, IEEE, ACM, SPIE 各会員。

中 澤 健 亮 (非会員) 1977 年生。2001 年 3 月 千葉工業大学工学部情報工学科卒業。2001 年 4 月 同大学大学院博士前期課程入学。主として画像処理の並列化による処理の高速化に関する研究に従事。

古 屋 穂 高 (非会員) 1978 年生。2000 年 3 月 千葉工業大学工学部情報工学科卒業。2002 年 3 月 同大学大学院博士前期課程修了。2002 年 4 月 (株) 日立製作所に入社。在学中は, 画像処理の並列化による処理の高速化に関する研究, 電力関係のシミュレーションの並列処理に従事。電子情報通信学会会員。