学術・技術論文

PCによる高速対応点探索に基づく ロボット搭載可能な実時間 視差画像・フロー生成法と実現

岡田 慧[*1]　加賀美 聡[*2]　稲葉 雅幸[*2]　井上 博允[*2]

Onbody Realtime Disparity Image and Flow Generation System based on Highspeed Correspondance Matching using PC

Kei Okada[*1], Satoshi Kagami[*2], Masayuki Inaba[*2] *and* Hirochika Inoue[*2]

This paper describes PC-based, robust, real-time disparity image and flow image generation system, which is able to be embedded in robot's body. Real-time feature is carried out in three steps: adopting recursive correlation method, optimizing algorithm to second CPU cache, utilizing MMX instruction set. Consistency checking method is adopted to achieve robustness, we integrate into recursive correlation loop to achieve realtimeness. Finally evaluation of proposed method is denoted.

Key Words: Realtime vision, Disparity image generation, Optical flow image generation, Recursive correlation, Robot vision.

1. はじめに

実世界で自律的に行動するロボットでは，環境への距離情報や，オプティカルフローによる環境の動きの情報といった視覚処理が欠かせない．従来はこれらの処理は，専用画像処理プロセッサを用いたアプローチ [1,2] やDSPによる専用ハードウエアを用いたアプローチ [3–5] により行われてきた．しかし多くの場合これらのシステムはさまざまな制約からロボットに搭載することが困難であり，ロボットによる環境情報の理解とそれに基づく高度な行動といった研究を行う妨げとなっている．

本論文は実世界で行動するロボットにとって重要な実時間かつロバストな視差画像，フロー画像生成がソフトウエアで可能であることを示す．これまでに，視差画像生成の高速化のために一次元の再帰相関演算法 [5,6] が画像処理の分野で知られているが，本論文では一次元及び新たに提案する二次元の再帰相関演算法，アルゴリズムの二次キャッシュへの最適化，マルチメディア命令の利用の三段階の高速化により，視差画像生成，フロー画像生成において単純に相関演算をソフトウエアで行うのに比べて汎用PCにおいて100倍程度の高速化が可能となった．また一貫性評価法を再帰相関演算手法のアルゴリズムの内部で行う手法を提案し，これによりロバスト性と実時間性を同時に達成する．最後に提案する手法の実時間性の評価実験結果を示す．

原稿受付　2000年2月2日
[*1]東京大学大学院工学系研究科情報工学専攻
[*2]東京大学大学院工学系研究科
[*1]Graduate School of Engineering, University of Tokyo
[*2]Faculty of Engineering, University of Tokyo

2. ロボット搭載可能な実時間視覚処理システム

視差画像，フロー生成に必要な対応点探索問題では，これまで主にエッジ等の特徴をマッチングする手法と，局所領域をマッチングする手法の二種類が研究されている．本論文では局所領域のマッチングを採用する．対応点探索問題は以下のように定式化できる．

(1) 画像のノイズ除去，正規化等の前処理を行う
(2) 各小領域に対して対応する各小領域ごとに相関演算を行う
(3) それらの中から適切な視差を選択する
(4) サブピクセルレベルで視差を推測する

実時間性を達成するために，最も計算量の多い (2) の処理を考える．従来からステレオ視の研究では，一次元の対応点探索において再帰相関演算手法を用いて計算量を減らす手法 [5,6] が利用されてきた．しかし，フロー生成では視差画像生と異なり二次元の対応点探索が必要になる．本論文では二次元の対応点探索の計算量を減らすことが可能な二次元再帰相関演算アルゴリズムを新たに提案し，実装により提案したアルゴリズムの有効性を示す．また汎用CPUへの実装の観点から，一次元，二次元の再帰相関演算手法におけるメモリ使用量の低減による二次キャッシュへの最適化，MMX命令による実装法を示す．

ロバスト性を達成するために (1) の前処理，(3) の視差の選択を考える．前処理ではLoGフィルターを，視差の選択では一貫性評価法による信頼度評価法 [7,8] を適用する．これらの処理は実時間性を満たす必要がある．本論文ではLoGフィルタをMMX命令により実装することで高速演算が可能であること，一貫性評価法を再帰相関演算ループに組み込むことで，メモリ使用量，計

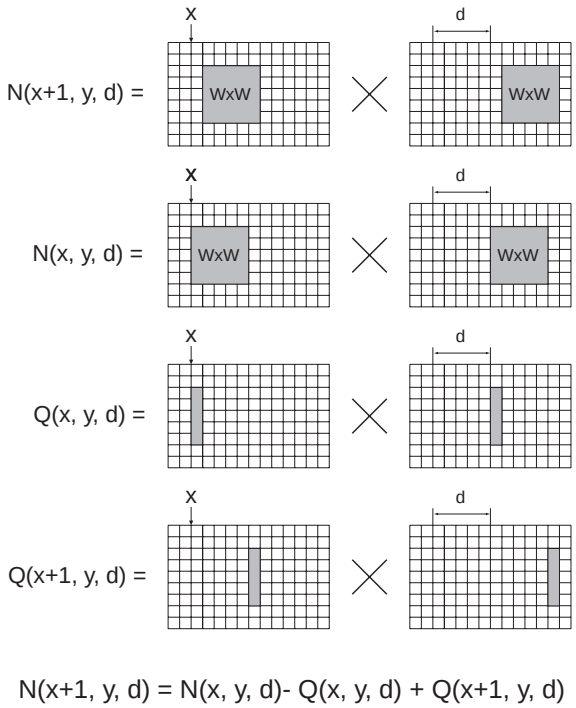

Fig. 1 Recursive correlation calculation: N(x+1,y,d) is able to calculated using N(x,y,d), Q(x,y,d) and Q(x+1,y,d). Q is also calculated using Q and P

算時間を増大させること無く実装する手法を示す.

ロボットに搭載可能するために，汎用 CPU を用いてシステムを構築した．汎用 CPU を用いたロボットシステム [9] は，コンパクトなシステム構築が可能，多種なデバイスが利用可能，蓄積されたソフトウェア資産が利用可能などの利点があり，近年注目されている．本論文ではこのロボットシステム上に視覚処理システムを構築することで，視覚処理を行いながら，環境を認識し行動を生成し，サーボループによりロボットの四肢を制御する，というロボットの知覚，認識，行動が可能な可能な自立／自律型ロボットシステムが可能なことを示す.

3. 再帰相関演算による対応点探索の高速演算法

3.1 相関演算による対応点探索

対応点探索では，片方の画像の各小領域に対して，もう一方の画像の対応する小領域を求める必要がある．各小領域の対応を計算するためには局所領域の相関値を求める相関演算を用う.

各画像の画素 (x,y) での輝度値を $I_1(x,y), I_2(x,y)$，その値域を $0{\leq}x,y{<}N$，相関演算のウインドウ i,j の値域を $0{\leq}i,j{<}W$，相関演算の探索範囲 d の値域を $0{\leq}d{<}D$ とする.

各小領域の輝度値の平均と分散が等しいとすると以下の式 (SAD:Sum of Absolute Difference) を用いることができる.

$$N(x,y,d) = \sum_{i,j} \left| I_1(x{+}i,y{+}j) - I_2(x{+}i{+}d,y{+}j) \right| \quad (1)$$

3.2 一次元再帰相関演算

視差画像生成の研究では，対応点探索において再帰相関演算手法を用いて計算量を減らす手法 [5,6] が利用されている．再帰相関演算を導入することで，視差画像生成における相関演算の計算量を $O(N^2W^2D)$ から， $O(N^2D)$ とすることが可能であることが知られている.

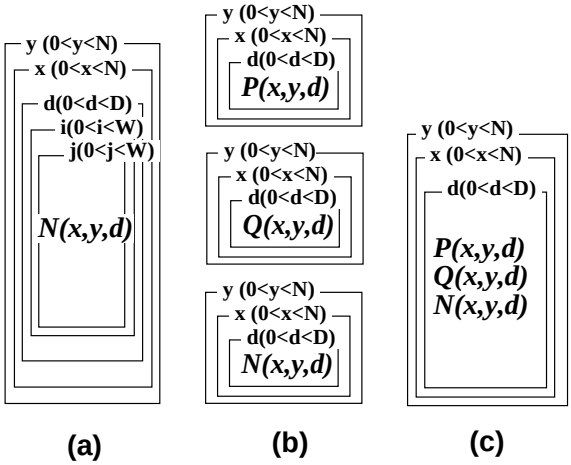

Fig. 2 Optimizing implementation with (a) normal correlation, (b) simple recursive correlation, (c) cache optimized recursive correlation. (Algorithms (b,c) don't work when x=0 or y=0)

SAD(式 **(1)**) による視差の計算は，以下になる.

$$\begin{aligned} O(x,y) &= \min_d\{N(x,y,d)\} \\ N(x,y,d) &= \textstyle\sum_{i,j} \left| I_1(x+i,y+j) - I_2(x+i+d,y+j) \right| \end{aligned} \quad (2)$$

ここで，入力画像の全ての画素に対し視差を計算する場合，$W \times W$ 回の乗算が冗長である．再帰相関演算はこの冗長性を排除する.

$$P(x,y,d) = \left| I_1(x,y) - I_2(x+d,y) \right|$$

とすると，式 **(1)** は

$$N(x,y,d) = \textstyle\sum_{i=0}^{W-1} \sum_{j=0}^{W-1} P(x+i,y+j,d)$$

と書ける．ここで，

$$Q(x,y,d) = \textstyle\sum_{j=0}^{W-1} P(x,y+j,d)$$

とすると，

$$N(x+1,y,d) = N(x,y,d) - Q(x,y,d) + Q(x+W,y,d)$$

と再帰的に計算できる (**Fig. 1**). さらに $Q(x,y,d)$ 自身も再帰的に計算できる．すなわち，以下のようになる.

$$\begin{aligned} P(x,y,d) &= |I_1(x,y) - I_2(x+d,y)| \\ Q(x,0,d) &= \textstyle\sum_j P(x,j,d) \\ Q(x,y+1,d) &= Q(x,y,d) - P(x,y,d) + P(x,y+W,d) \\ N(0,y,d) &= \textstyle\sum_i Q(i,y,d) \\ N(x+1,y,d) &= N(x,y,d) - Q(x,y,d) + Q(x+W,y,d) \end{aligned} \quad (3)$$

```
unsigned char   I1[N][N], I2[N][N];
unsigned short  P[W][N][D], Q[2][N][D], R[N][D];
asm volatile("pxor %%mm7,%%mm7"::);                          /* mm7 = 0 */
asm volatile("movd %0,%%mm6"::"m"(I1[y][x]));                /* mm6 = I1 */
asm volatile("punpcklbw %%mm7,%%mm6"::);
for (d=0; d<D; d+=4) {
  asm volatile("movd %0,%%mm1"::"m"(I2[y][x+d]));            /* mm1 = I2 */
  asm volatile("punpcklbw %%mm7,%%mm1"::);
  asm volatile("movq %%mm6,%%mm2"::);                        /* mm2 = mm6 */
  asm volatile("movq %%mm1,%%mm0"::);                        /* mm0 = mm1 */
  asm volatile("psubusb %%mm2,%%mm0"::);                     /* mm0 = mm0 - mm2 */
  asm volatile("psubusb %%mm1,%%mm2"::);                     /* mm2 = mm2 - mm1 */
  asm volatile("por %%mm0,%%mm2"::);                         /* mm2 = mm2 | mm0 */
  asm volatile("movq %%mm2,%0":"=m"(P[y%W][x][d]));          /* P   = mm2 */
  asm volatile("movq %0, %%mm3"::"m"(Q[(y-1)%2][x]));        /* mm3 = Q */
  asm volatile("paddusw %%mm2, %%mm3"::);                    /* mm3 = mm3 + mm2 */
  asm volatile("psubusw %0, %%mm3"::"m"(P[(y-W+1)%W][x][d])); /* mm3 = mm3 - P */
  asm volatile("movq %%mm3, %0":"=m"(Q[y%2][x]));            /* Q   = mm3 */
  asm volatile("paddusw %0, %%mm3"::"m"(R[x-1]));            /* mm3 = mm3 + R */
  asm volatile("psubusw %0, %%mm3"::"m"(Q[y%2][x]));         /* mm3 = mm3 - Q */
  asm volatile("movq %%mm3, %0":"=m"(R[x]));                 /* R   = mm3 */
}
```

Fig. 3 MMX implementation of Equation (3). Absolute difference is impremented using 2 sub instruction and 1 or instruction to avoid using branch jump.

3.3 二次元再帰相関演算

フロー生成では視差画像と異なり二次元の対応点探索が必要になる．本節では二次元再帰相関演算アルゴリズムを新たに提案する．このアルゴリズムにより，フロー生成の計算量を $O(N^2W^2D^2)$ から，$O(N^2D^2)$ に減らすことが可能である．

フローの計算は，以下のように書くことができる．

$$N(x,y,k,l) = \sum_{i,j} \begin{aligned}&|I_1(x+i,y+j)\\ &-I_2(x+i+k,y+j+l)|\end{aligned} \quad (4)$$

ここでは視差画像生成と同様に $W \times W$ 回の乗算が冗長である．すなわち，$N(x,y,k,l)$ は以下のように再帰的に計算できる．これを我々は二次元再帰相関演算アルゴリズムと呼ぶ．

$$\begin{aligned}
P(x,y,k,l) &= \left|I_1(x,y) - I_2(x+k,y+l)\right| \\
Q1(x,0,k,l) &= \textstyle\sum_j P(x,j,k,l) \\
Q1(x,y+1,k,l) &= Q1(x,y,d) \\
&\quad +P(x,y+W,k,l) - P(x,y,k,l) \\
Q2(0,y,k,l) &= \textstyle\sum_i P(i,y,k,l) \\
Q2(x+1,y,k,l) &= Q2(x,y,d) \\
&\quad +P(x+W,y,k,l) - P(x,y,k,l) \\
\\
N(0,y,k,l) &= \textstyle\sum_i Q1(i,y,k,l) \\
N(x,0,k,l) &= \textstyle\sum_j Q2(x,j,k,l) \\
N(x+1,y+1,k,l) &= N(x,y,k,l) \\
&\quad +Q1(x+W,y,k,l) - Q1(x,y,k,l) \\
&\quad +Q2(x,y+W,k,l) - Q2(x,y,k,l) \\
&\quad +P(x,y,k,l) + P(x+W,y+W,k,l) \\
&\quad -P(x+W,y,k,l) - P(x,y+W,k,l)
\end{aligned} \quad (5)$$

3.4 キャッシュへの最適化と MMX 命令を用いた実装

本節では汎用 CPU への実装の観点からメモリ使用量の低減による二次キャッシュへの最適化，MMX 命令による実装法を示す．本論文での議論は視差画像生成を対象とするが，フロー生成にも適用できる．

3.4.1 メモリ使用量の低減によるキャッシュへの最適化

Fig. 2 は (a) 視差の計算 (式 **(2)**) を単純に実現したもの，および (b) 再帰相関演算 (式 **(3)**) を単純に実現した際のループの構成法を示している．全画面の対応点を計算するためには，再帰相関演算の式を x, y, d の 3 重のループとして計算することになるが，単純に式の通りに実装するとキャッシュを利用できず，計算速度は CPU の演算速度ではなくメインメモリの読み出し／書き込み速度に依存することになってしまう†．

そこでキャッシュの利用効率を最大にするために演算途中に使用する P,Q を計算する際に N も同時に計算することとする (**Fig. 2** (c))．これにより単純にインプリメントすると P,Q,N は $N\times N\times D$ の配列サイズをそれぞれ持つことになっていたが，この場合には P は $N\times W\times D$，Q は $N\times 2\times D$，N は $N\times D$ の配列サイズで良いことになる．

従って，本論文で述べたアルゴリズムに必要なメモリ領域は，N=128, W=16, D=32，画像の各ピクセルのデータ量を 8bit とすると，P は 8bit, Q,N は 16bit となることから，約 2.6MB となるが，この方法により約 85KB まで低減することができ，Pentium の 2 次キャッシュに入りきる．実際にはこれ以外に画像領域として $128\times128\times2\times8bit$= 32KB が必要である．

3.4.2 MMX 命令を用いた実装

MMX 命令は Intel の PentiumMMX プロセッサ以降に搭載されたマルチメディア命令で，一クロックで 64bit 長の MMX レジスタ上（あるいはメモリ上）におかれたデータ配列の和，積，積和，論理演算等を行う，いわゆる SIMD (Single Instruction Multi Data) な命令セットで，積以外は一クロックで実行され

† メインメモリからのデータの読み出し，書き込みは 10 クロック以上かかるが，2 次キャッシュからは 1 または 2 クロックで行える

る．ただし割り算命令をもっていないことと，積は 16bit × 16bit までしかできないといった制約がある．

式 **(3)** のは MMX 命令を用いて **Fig. 3** のように実現される†．ここでは 8bit 整数の画像配列をレジスタに移した後，16bit 整数に拡張し，4 つずつまとめてお互いに差をとり，2 つの結果を or することにより条件分岐を行わずに差の絶対値を計算し，P に代入している．このコードは C 言語で書いたものに比べて約 2 倍早く実行される．

4. 一貫性評価法を用いた信頼度評価とその実装法

ロボットの視覚ではロバスト性，すなわち特徴量が少なかったりオクルージョンが存在する環境でも信頼のおける視差画像を生成する必要がある．そのために，対応点探索結果の信頼度を評価する機構を組み込むことで，処理結果を利用する際に信頼度を評価できることが必要になる．

この問題を解決するために様々な研究が行われているが，本論文では対応点探索における相関演算の信頼度評価に”Left-to-Right Right-to-Left Consistency Checking(一貫性評価法)”を用いる [7,8].

4.1 一貫性評価法

一貫性評価法は片側の画像の点に対応する反対側の画像上の点からの対応点が，最初の点に一致する場合のみ，その対応が信頼できるとするものである．この方法により，対応点がとれない場所がはっきり区別出来るようになる．アルゴリズムは以下の通りである．

(1) 右画像中の局所領域を参照領域 (領域 1) とし，左画像を探索し最も相関の高い局所領域を領域 1 に対応する領域 (領域 2) とする．
(2) 領域 2 を参照領域とし，右画像を探索し最も相関の高い局所領域を領域 2 に対応する領域 (領域 3) とする．
(3) 領域 1 と領域 3 が同じ局所領域であれば，領域 1 と領域 2 は左右画像中の対応する領域とする．

4.2 一貫性評価法の再帰相関演算への組み込み

一貫性評価法を単純に導入すると右画像から左画像の対応点探索，左画像から右画像への対応点探索と一貫性評価法を導入しない場合に比べ二倍の対応点探索を行う必要がある．ここで対応点探索の処理は，「各小領域に対して対応する各小領域ごとに相関演算を行う」，「それらの中から適切な視差を選択する」の 2 段階の処理からなることに注意すると，相関演算は右画像から左画像，左画像から右画像と二度行う必要は無く，相関演算は一度おこない得られた相関値分布を再配列し視差の選択を二度おこなえばよい．

従って一貫性評価法は **Fig. 2** (c) の y のループ内の最後に x 方向の一ライン分まとめて行うことにより，計算時間は 30% 程度増加する程度で行える．

5. ＰＣを用いたロボット搭載用画像処理システムの実現

†gcc(gas) の記述のためにオペランドのソースとデスティネーションは通常の x86 の方式と逆になっていることに注意されたい．%%mm0,1,2 は 8 個存在する MMX レジスタを指す

Table 1 Realtime Performance of Disparity Image Generation

Cost Function	PentiumIII 700MHz		
	without CC	with CC	sub-pixel
a) Simple Correlation	1312.2	2915.3	2624.9
b) Recursive Correlation	27.7	41.2	43.3
c) Cache Optimal Correlation	21.2	27.1	29.0
d) MMX Imprementaion	13.4	18.4	20.9

(unit:msec) (N=128, W=13, D=32)

Table 2 Realtime Performance Result of Optical Flow Generation

Cost Function	PentiumIII 700MHz		
	without CC	with CC	sub-pixel
a) Simple Correlation	1656.9	3013.0	2925.5
b) Recursive Correlation	65.4	78.3	81.0
c) Cache Optimal Corrleation	33.5	42.3	49.2
d) MMX Implementation	19.8	27.2	32.6

(unit:msec) (N=80, W=15, D=8)
Row : a) Simple Correlation, b) Recursive Correlation, c) Cache Optimal Corrleation, d) MMX Implementation.
Column : 1)Without Consistency Checking, 2) With Consistency Checking, 3) With Consistency Checking and Sub Pixel Interpolation.

Fig. 4 Left Top: Input image, Right Top: Disparity image (brighter is closer), Left Buttom: Disparity image with consistency checking, Right Buttom: Disparity image with consistency checking and sub-pixel interpolation

5.1 ロボット搭載用画像処理システム

本節では汎用 CPU を用いたロボット搭載可能な画像処理システムの構築法を示す．近年ロボットに PC と視覚処理ボードを搭載し画像処理を行いながら行動する知能ロボットの研究が盛んに行われている [9,10]．これらのロボットシステムは，PC 用に開発された多種なデバイスが利用可能，蓄積されたアプリケーション，開発環境が利用可能，コンパクトなシステム化による自立型ロボットを構築可能といった特徴がある．汎用 CPU を用いた画像処理システムをこのロボットシステム上に構築することで，視覚処理を行いながら，環境を認識し行動を生成し，サーボループ

Fig. 5 Real-time optical flow generatio experimen. Top: Input image, Middle: Optical flow, Buttom: Optical flow with sub-pixel interpolation.

にロボットの全身を制御する，というロボットの知覚，認識，行動が可能な可能な自立型ロボットシステムが可能になる．

また，視差画像生成システムでは，ステレオカメラからの画像の取り込みが重要な問題となってくる．ここではロボットに搭載することを目的として本研究室で開発された，同期して動作している 2 つのカメラの映像信号を電子的にマージし，インターレースされたステレオ画像を取り込み装置に与える方法を採用した [11].

画像取り込み装置は Bt848 という画像取り込み IC を用いたキャプチャーカードを用いた．このハードウエアはリアルタイムで画像をメモリに DMA 転送する機能を持ち， Linux 用のデバイスドライバが公開されていることから，容易に実時間画像処理システムを構築することができる，という特徴を持っている．

5.2 前処理，及び後処理の実装

実際のロボットシステムでは対応点探索の際に，画像のノイズの除去や正規化等の前処理，及びサブピクセルレベルでの視差の推定の後処理が必要になる．これらの処理を本論文で述べてきたシステムに組み込む．

5.2.1 LoG フィルタによる前処理の MMX による実装

画像のノイズ除去，正規化に効果のある LoG フィルタを MMX により実装した． MMX による実装ではレジスタ間のデータ依存に注意することで， PentiumIII-700MHz を用いた場合 $2.91msec$ で処理できる．この時のフィルタの大きさは 7×7 であり，画像の大きさは 128×128 であった．

5.2.2 サブピクセルレベルの視差の推定の実装

二次補間に基づくサブピクセル法を適用した．視差画像におけるサブピクセルの補正項を $\overline{d}$ は以下の式で求める．

$$\overline{d} = \frac{1}{2} \frac{N(x,y,d-1) - N(x,y,d+1)}{N(x,y,d-1) + N(x,y,d+1) - 2N(x,y,d)}$$

ここで， N(x,y,d) は画素 (x, y) と画素 $(x + d)$ との相関値である．これを用いることで，視差は $d + \overline{d}$ と推定される．フローの場合も同様である．

5.3 実験

本手法の実時間性を評価した．実験は PentiumIII-700MHz(外部バス 100MHz) を用いて linux 2.2.13 上で行った． C 言語は全て gcc-2.7.2.3 を用いて，また MMX は gcc 上でインラインアセンブラを用いて実現された．

Table. 1 に視差画像生成の処理時間を， **Table. 2** にフロー生成の処理時間を示す．表は a) SAD による対応点探索による視差計算 (式 **(2)**) を単純に実装した場合， b) 再帰相関演算 (式 **(3)**) を単純に実装した場合， c) キャッシュを意識した再帰相関演算を実装した場合， d) MMX を用いた実装，のそれぞれのバージョンのソフトウエアについて， 1) 信頼度評価を行わない場合， 2) 一貫性評価法による信頼度評価を行った場合， 3) 信頼度評価を行いサブピクセル法により視差を推定した場合の計算時間を計測したものを示す．

視差画像生成の処理結果を **Fig. 4** に示す．実験は屋外環境で行った．ロボットは研究室で開発された JROB-1 を用いた．画像処理，行動生成がロボットに搭載された PC で処理できるため，屋外での行動が実現できる．左上がカメラで得られた画像，左下が視差画像 (サブピクセル無し)，右下が視差画像 (サブピクセル有り) である．白い部分がカメラに近い領域であり，障害物となる人間が認識できる．これに基づきロボットは自律的な障害物回避行動を実現した [12]．パラメータは $N = 128, W = 13, D = 32$ である．

フロー生成の処理結果を **Fig. 5** に示す．左列が入力画像，右列上がオプティカルフローを計算し一貫性評価法を適用した出力，右列下がオプティカルフローを計算し一貫性評価法を適用し，さらにサブピクセル法に基づきオプティカルフローを推定した出力である．パラメータは $N = 64, W = 9, D = 8$ である．

6. おわりに

本論文では，実世界で自律的に行動するロボットにとり重要な視差画像，フロー生成を実時間で行うための視覚処理生成システムの構築法を示した．ロボット視覚では実時間性，ロバスト性，ロボット搭載可能性が重要になる．実時間性を達成するために，従来から視差画像生成の研究では再帰相関演算を用いる手法が知られている．本論文では二次元再帰相関演算手法を提案しフロー生成に適用した．また，汎用CPUへ実装の観点からメモリ使用量を低減によるキャッシュへの最適化法，MMX命令による実装法を示す．また，ロバスト性を達成するために一貫性評価法による信頼度評価法を導入した．その際にロバスト性と共に実時間性を達成することが重要になる．本論文は，一貫性評価法を再帰相関演算ループに組み込むことで，メモリ使用量，計算時間を増大させること無く実装する手法を示す．これらの手法を用いてロボット搭載可能な実時間性，ロバスト性を有する視覚処理システムを構築した．

本論文で示したシステムにより距離，動きなどの情報を視野全体で密に獲得できる．これをロボットに搭載することで，ロボットは豊富な情報を獲得でき，家庭，オフィス等の実環境における，自律的な行動を実現する環境が整うと考える．さらに，本論文ではロボットへの搭載を指向したシステム構築を行った．従来の画像処理システムでは，ロボット全体のシステムを統合するという観点が見落とされることもしばしばあり，ロボットへの利用が困難なもも多い．当研究室ではすでに，車輪型ロボット [13]，四脚型ロボット [12]，上半身人間型ロボット [14] などで利用されている．

参考文献

[1] H. Inoue, T. Tachikawa, and M. Inaba. Robot Vision System with a Correlation Chip for Real-time Tracking, Optical Flow and Depth Map Generation. In *Proc. IEEE International Conference on Robotics and Automation*, pp. 1621–1626, 1992.

[2] 内山隆, 沢崎直之, 青木孝, 森田俊彦, 稲本康, 佐藤雅彦, 稲葉雅幸, 井上博允. ビデオレートトラッキングビジョンの実用化. 第12回日本ロボット学会学術講演会予稿集, pp. 345–346, 1994.

[3] 塩原守人, 江川宏一, 佐々木繁. リアルタイム・オプティカル・フロー・プロセッサ ISHTAR. 画像の認識・理解シンポジウム, MIRU94-Vol.2, pp. 295–302, Aug. 1994.

[4] S. Rougeaux and Y. Kuniyoshi. Velocity and Disparity Cues for Robust Real-Time Binocular Tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–6, 1997.

[5] 金出武雄, 蚊野浩, 木村茂, 川村英二, 吉田収志, 織田和夫. ビデオレートステレオマシンの開発. 日本ロボット学会誌, Vol. 15, No. 2, pp. 261–267, Mar. 1997.

[6] O. Faugeras, B. Hots, H. Mathieu, T. Viéville, Z. Zhang, P. Fua, E. Théron, L. Moll, G. Berry, J. Vuillemin, P. Bertin, and C. Proy. Real Time Correlation-Based Stereo: Algorithm, Implementations and Applications. Technical Report N°2013, INRIA, 1993.

[7] R. Bolles and J. Woodfill. Spatiotemporal Consistency Checking of Passive Renge Data. In T. Kanade and R. Paul, editors, *Robotics Research: The Sixth International Symposium*, pp. 165–183. International Foundation for Robotics Research, 1993.

[8] P. Fua. A Parallel Stereo Algorithm that Produces Dense Depth Maps And Preserves Images Features. In *Machine Vision and Applications*, pp. 35–49, 1991.

[9] 加賀美聡, 桃澤光隆, 岡田慧, 松本吉央, 近野敦, 稲葉雅幸, 井上博允. 脚型ロボット感覚行動統合研究プラットフォーム JROB–1. 日本ロボット学会誌, Vol. 16, No. 5, pp. 47–52, 1998.

[10] 松本吉央, 坂井克弘, 稲邑哲也, 稲葉雅幸, 井上博允. PC ベースのハイパーマシン. 第15回日本ロボット学会学術講演会予稿集, pp. 979–980, 1997.

[11] Y. Matsutmoto, T. Shibata, K. Sakai, M. Inaba, and H. Inoue. Real-Time Color Stereo Vision System for a Mobile Robot based on Field Multiplexing. In *In Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 1934–1939, 1997.

[12] K. OKADA, S. KAGAMI, M. INABA, and H. INOUE. Vision-based action control of quadruped legged robot jrob-1. In *Proc. of 9th International Conference on Advanced Robotics(ICAR'99)*, 1999.

[13] 稲邑哲也, 佐川立昌, 稲葉雅幸, 井上博允. 照明条件の変動下における人間の発見追従行動のための実時間視覚処理. 第16回ロボット学会学術講演会予稿集, 第3巻, pp. 1039–1040, 1998.

[14] 加賀美聡, 近野敦, 陰山竜介, 桃澤光隆, 稲葉雅幸, 井上博允. 視触覚行動統合研究のための車輪移動上半身型ヒューマノイドH4の設計と開発. 第16回ロボット学会学術講演会予稿集, 第2巻, pp. 835–836, 1998.

岡田 慧 (kei OKADA)

1974年2月24日生まれ．1997年京都大学工学部情報工学科卒業．1999年東京大学大学院工学系研究科情報工学専攻修士課程修了．現在，同博士課程に在籍．実時間視覚処理の研究に従事．

（日本ロボット学会学生会員）

加賀美 聡 (satoshi KAGAMI)

1970年3月14日生まれ．1992年上智大学理工学部機械工学科卒業．1997年東京大学大学院工学系研究科情報工学専攻博士課程修了．博士（工学）．現在，日本学術振興会未来開拓学術研究推進事業「マイクロ・ソフトメカトロニクス統合体としての高度生体機能機械の研究」プロジェクトのリサーチアソシエート．

（日本ロボット学会正会員）

稲葉 雅幸 (masayuki INABA)

1958年5月23日生．1986年東京大学大学院工学系研究科情報工学専門課程博士課程修了．工学博士．1986年東京大学講師，1989年助教授，2000年東京大学教授大学院工学系研究科機械情報工学専攻，現在大学院情報学環・学際情報学府所属．日本機械学会，計測自動制御学会，情報処理学会，人工知能学会，ソフトウェア科学会等会員．

（日本ロボット学会正会員）

井上 博允 (hirochika INOUE)

1942年7月生.1965年東京大学工学部産業機械工学科卒業．1970年同大学院博士課程修了．工学博士．同年電子技術総合研究所入所.1977年東京大学工学部機械工学科助教授.1984年教授．現在，機械情報工学科教授．ロボット全般，人工知能，情報システム工学の研究と教育に従事．

（日本ロボット学会正会員）