社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2006－SLDM－127 (33)
2006／11／30

Cベース設計による粒子追跡技術のリアルタイム化

上甲 憲市‡　上津 寛和‡

中川 寛誉†　神戸 尚志†

‡近畿大学大学院 総合理工学研究科 〒577-8502 大阪府東大阪市小若江3-4-1

†近畿大学 理工学部 電気電子工学科 〒577-8502 大阪府東大阪市小若江3-4-1

E-mail: ‡0533340409y@kindai.ac.jp, tkambe@ele.kindai.ac.jp

あらまし 粒子追跡技術は、物体や流体の微細な変形や運動を画像計測する上で重要な技術であるが、ソフトウェア処理では多大な計算時間を要する。粒子の画像をリアルタイムで解析することを目的とし、C言語で書かれた粒子追跡法(PTV)のプログラムを、Bachシステムを用いてハードウェア化を行い、高速化を図る。

キーワード 粒子マスク相関法，ハードウエア/ソフトウエア協調設計，Bachシステム

C-Base Design of a Particle Tracking System

Kenichi JYOKO‡　Hirokazu UETSU‡

Hirotaka NAKAGAWA†, and Takashi KAMBE†

† ‡ Kinki University, 3-4-1 kowakae, Higashi-Osaka City, Osaka, 577-8502 Japan

E-mail: ‡0533340409y @kindai.ac.jp, tkambe@ele.kindai.ac.jp

Abstract Particle tracking technology is one of the most promising methods for the measurement of the velocity fluctuation of flows. This paper describes the design of a software and hardware system based on the particle mask correlation (PMC) method and the KC method. The computational overhead is accelerated by the use of secondary differentiation preprocessing, 1-pixel processing, cache memory utilization, area limitation of the particle search. The processing speed, the circuit size of the system are evaluated.

Keyword Particle Mask Correlation method, software/hardware co-design, Bach system

1. はじめに[2]

粒子追跡技術は、ある時間間隔で画像中の各トレーサ粒子の移動を自動的に追跡し、流れ場を計測する。この技術を応用し、物体や流体の微細な変形や運動を画像計測することができる。応用例として流体観測できなかった種々のミクロ・スケールの高速現象、ハードディスク等の高速回転体やそれに付随する高速流れによって生じる境界層の構造の乱れ、微細掘削加工時に生じる様々な高速現象等を立体的スローモーションで観察することが可能になると考えられる。図1.1に粒子追跡の様子を示す。

本研究では、まずC言語で記述された粒子追跡プログラムを解析し、計算時間を要している部分に対してハードウェア化を行い、リアルタイム処理(15fps)を目標に高速化を図る。

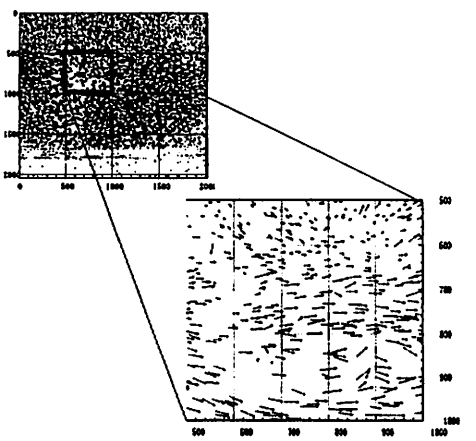


図1.1 粒子追跡の様子

2. 粒子追跡システムとハードウェア化

2.1. 粒子追跡システムの概要[3,4]

粒子追跡システムは粒子マスク相関法とKC法で構

成される。粒子マスク相関法(以下 PMC 法) とは粒子画像から円形に近い粒子の抽出を、相関値計算を用いて行う手法である。粒子の理想的なテンプレート画像(以下マスクと呼ぶ)を用いて粒子画像上を走査し、相関係数を計算する。今回のマスク画像のサイズは７×７とする。相関係数 r は-1<r<1 の範囲をとる。マスクと、走査位置での輝度分布の形状が等しい程、輝度値は異なっていても相関係数は 1 に近づく。相関係数があるしきい値以上になる部分に粒子が存在するとみなし、粒子の位置を割り出し、粒子位置情報（粒子の個数、粒子の X 座標、粒子の Y 座標、粒子の大きさ）を出力する。図 2.1 に PMC 法におけるマスクとその走査の様子を示す。

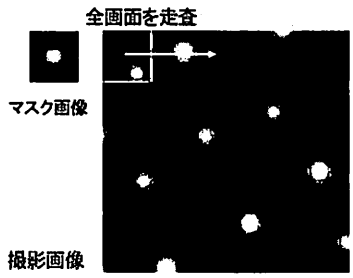
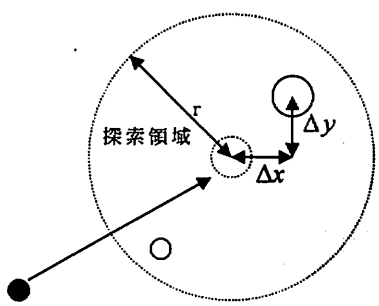


図 2.1 PMC 法による粒子位置検出

KC 法は、確立・統計学の基本に基づいたカルマンフィルタ理論を粒子追跡に適用し、x^2 検定でどの予測粒子がどの実測粒子に対応しているかを求める手法である。



カルマンフィルタによる予測

○ :（ｔ＋１）時刻の実測粒子位置
◌ :カルマンフィルタによる（ｔ＋１）時刻の実測粒子位置
● : ｔ時刻の粒子位置

図 2.2 カルマンフィルタ・x^2検定法の概念図

ある時刻 t の粒子座標、速度、加速度などの状態量がわかっていれば、次時刻 t+1 の予測値がカルマンフィルタによって推定される。その予測値と観測値から、x^2 検定によって同一粒子の同定を行う。同定ができれば、観測データからさらに t+2 時刻の予測値が求められ、さらに同様の操作を繰り返し行うことにより同一粒子を時々刻々追跡していく。

図 2.2 を用いて同定方法を説明する。時刻 t に黒点●の位置にあった粒子が t+1 には点線の○の位置に移動すると推定されたとする。対応する粒子候補は点線の大円内の大小２個の○で示されている。このとき、対応する粒子候補すべてと x^2 検定を行い、値が最小値となるものを同一粒子と対応付けをする。

2.2. 粒子追跡システムのアルゴリズム

粒子追跡システムのアルゴリズムについて説明する。①から⑤の PMC 法と⑥から⑦の KC 法の処理を行うことによって粒子抽出を行う。また、図 2.2 にカルマンフィルタ・x^2 検定法の概念図を示す。

① 画像拡張：粒子画像から読み込んだデータを 4 倍に拡大する。これにより抽出精度を高める。
② 相関値計算:マスク画像を 1 枚の静止画像に置き、マスク画像と粒子画像の相関係数を求める。相関値計算式を（2.1）式に示す。
③ 二値化処理：相関係数がしきい値 0.7 以上の領域とそれ以下の領域に二値化する。二値化は粒子像を抽出するための処理であり、背景の輝度値が０の場合、輝度値１をもつ画素の一つの集まりは基本的に１個の粒子像に対応する。
④ 連結領域の抽出（ラベリング）:粒子画像中には粒子が多数存在する。個々の粒子を区別するためそれぞれに番号付けを行う。
⑤ 粒子像中心座標計算：各粒子ごとに粒子の中心を求め、その座標を粒子位置とする。
⑥ カルマンフィルタ：カルマンフィルタにより時刻 t 画面上にある粒子の(t+1)画面上における位置を推定する。
⑦ x^2 検定:(t+1)画面上において推定された粒子位置から半径 r の検索領域以内にある粒子すべてについて x^2 検定を行う。x^2 値が最小となるものを仮の対応する粒子とし、x^2 値がある棄却水準以下であれば同一粒子と確定する。棄却水準以上であれば対応する粒子がないものとする。

$$r=\frac{\sum_{i,j=1,1}^{n,m}\left(I_{i,j}-\bar{I}\right)\left(M_{i,j}-\bar{M}\right)}{\sqrt{\left[\sum_{i,j=1,1}^{n,m}\left(I_{i,j}-\bar{I}\right)^2\right]\left[\sum_{i,j=1,1}^{n,m}\left(M_{i,j}-\bar{M}\right)^2\right]}} \quad (2.1)$$

n,m:水平,鉛直相関領域サイズ　i,j:水平,鉛直位置
I :元画像の輝度値　$\bar{I}$:元画像の輝度値の平均
M :マスク画像の輝度値の平均 $\bar{M}$:マスク画像の平均値

2.3. リアルタイムシステムの設計

PMC 法は Imread 関数(画像読み込み、画像拡張)、Cor

関数(相関値計算、二値化)、Chain 関数(ラベリング、粒子像中心計算)で構成され、KC 法はカルマンフィルタと x^2 検定からなる。

リアルタイム処理を実現するために全処理のハードウェア化を行い、各処理の高速化を行う。設計するシステムの概要を図 2.3 に示す。各モジュールを 66.66[ms]で動作させれば、1 枚の画像の処理には 266.64[ms]の時間がかかるが、パイプラインを 66.66[ms]単位で行い、リアルタイム処理が実現可能となる。

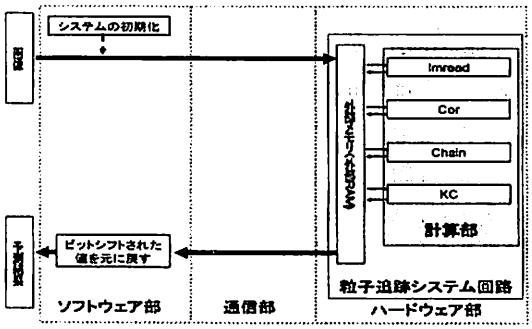


図 2.3 システムの概要

2.4. ハードウェア化技術

現在の HDL によるハードウェア設計は、C 言語等を用いてアルゴリズム設計・検証を行い、その後、HDL での RT レベル設計、論理合成、シミュレーションによる検証という手順で行われる。しかし、HDL での RT レベル記述とプログラミング言語との差は大きく、HDL で回路設計を行った後、不具合や変更によるアルゴリズムの修正には、大きな手間が必要となる。Bach システム[1]を用いた設計では、トップレベルの設計から抽象度の高い Bach-C を用い、機能設計・検証を行うことで、HDL による記述・検証期間が削減されるだけでなく、Bach-C 記述から RT レベルの VHDL が自動生成され、ハードウェアの設計が大幅に効率化される。それにより短期間に様々なアーキテクチャを設計できる。

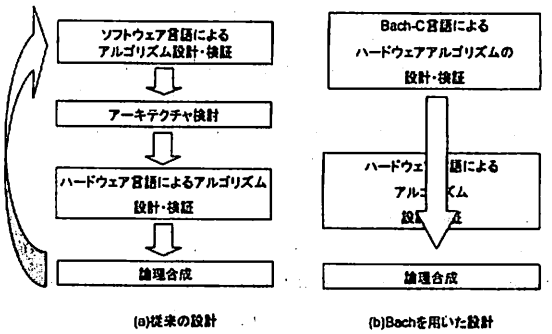


図 2.4 Bach システムによる設計工程

3. 相関値計算式の高速化

相関値計算部について以下の 7 種類の高速化を行った。

(A) 二次微分による粒子判別
(B) 相関値判別式の変形
(C) データ再利用によるメモリアクセス数の削減とレジスタ追加によるメモリアクセスの削減
(D) 相関値判別式の演算回路化
(E) 画像分割による並列化

以下、各々の処理について詳細を述べる。

(A) 二次微分による粒子判別

粒子画像中、明らかに背景である画素については相関値計算を省くことで処理を高速化する。

特定の画素に対して、粒子の一部である可能性があるのか、明らかに背景であるのかを判断するため、注目画素を中心に前後の画素との輝度の二次微分値を用いる。この値は、粒子のような輝度値が盛り上がった場所において負の大きな値をとる。この値を検出することで、多くの背景画素において相関値計算を省くことができる。二次微分値を求める式を(3.1)式に示す。

$$(f(x+d)-f(x))-(f(x)-f(x-d)) = f(x+d)-2f(x)+f(x-d) \quad (3.1)$$

x:注目画素の画像上の x 軸位置
$f(x)$:位置 x の輝度値　d:注目画素からの距離

実際の粒子画像について二次微分値を算出した結果を図 3.1 に示す。粒子の輝度値の盛り上がりは 3 箇所確認できるが、それぞれにおいて二次微分値は負の値をとっていることがわかる。

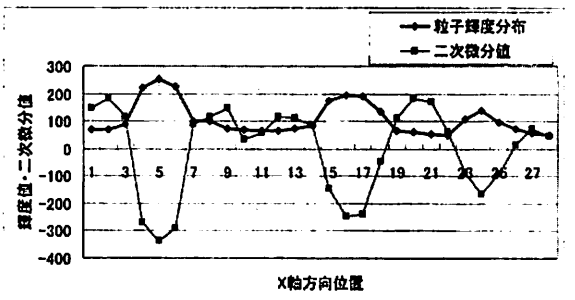


図 3.1 実写画像輝度分布とその二次微分値
(前後幅 7 画素)

X 軸方向 1 方向について述べたが、これを Y 方向にも適用することで、より多くの背景における相関値計算を削減できる。

(B) 相関値判別式の変形

式 2.1 の相関値判別式では平均値計算や除算が必要となり、処理時間や回路規模がともに増大する。そのため式 2.1 を以下の式 3.2 のように変形し、平均値計算を削除する[5]。

$$0.7 < \frac{n\times m\sum_{i,j=1,1}^{n,m} I_{i,j}M_{i,j} - \left(\sum_{i,j=1,1}^{n,m} I_{i,j}\right)\left(\sum_{i,j=1,1}^{n,m} M_{i,j}\right)}{\sqrt{n\times m\sum_{i,j=1,1}^{n,m} I_{i,j}^2 - \left(\sum_{i,j=1,1}^{n,m} I_{i,j}\right)^2}\sqrt{n\times m\sum_{i,j=1,1}^{n,m} M_{i,j}^2 - \left(\sum_{i,j=1,1}^{n,m} M_{i,j}\right)^2}} \quad (3.2)$$

式 3.2 から平方根計算と除算計算をなくすため移項などを行い、式 3.3 のように変形することによって計算の高速化、回路規模の削減を行う。

$$(0.7)^2\left(n\times m\sum_{i,j=1,1}^{n,m} I_{i,j}^2 - \left(\sum_{i,j=1,1}^{n,m} I_{i,j}\right)^2\right)\left(n\times m\sum_{i,j=1,1}^{n,m} M_{i,j}^2 - \left(\sum_{i,j=1,1}^{n,m} M_{i,j}\right)^2\right) \quad (3.3)$$
$$< \left(n\times m\sum_{i,j=1,1}^{n,m} I_{i,j}M_{i,j} - \left(\sum_{i,j=1,1}^{n,m} I_{i,j}\right)\left(\sum_{i,j=1,1}^{n,m} M_{i,j}\right)\right)^2$$

(C) データ再利用によるメモリアクセス数の削減

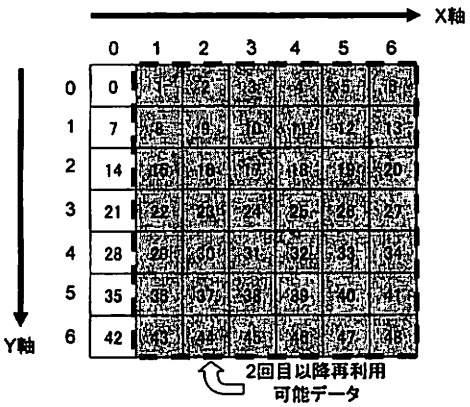


図 3.2 レジスタに保持する画像データ

マスク画像サイズが 7×7 画素であるため、元のプログラムでは 49 回メモリにアクセスして画像データを取得していた。しかし、2 回目以降はマスク画像を右に 1 画素ずらし、次の相関値計算を行うので図 3.2 に示すレジスタで保持されているデータの X=0 列以外のデータ 42 箇所の画像データを再利用することで、メモリアクセス数を削減する。処理の流れを図 3.3 に示す。

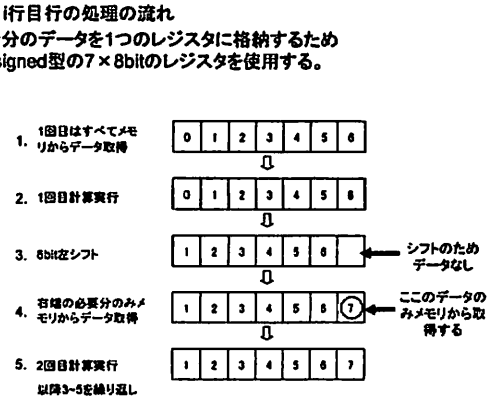


図 3.3 データ再利用によるメモリアクセス数の削減の処理の流れ

また、図 3.4 の Y=7 行目のように 1 行分の画像データを格納するレジスタを追加することによって 2 回の相関値計算を行うことができる。さらに X=7 列の 8 箇所のメモリアクセスのみで 3，4 回目の相関値計算を行うことができ、メモリアクセス数の削減を行うことができる。

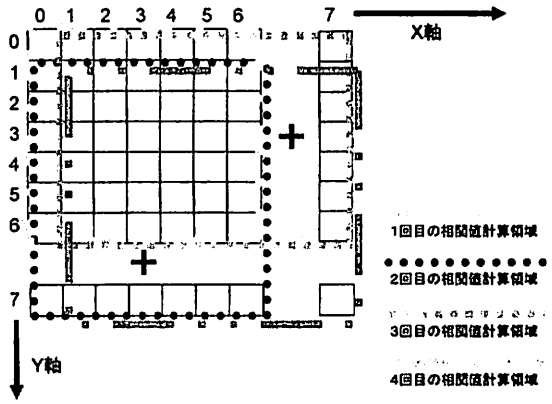


図 3.4 レジスタ追加によるメモリアクセスの削減

(D) 相関値判別式の演算回路化

Bach システムでは合成時に#pragma preserve を記述することによって関数を組み合わせ回路の演算器として扱うことができる。そのため、相関値計算を行う関数に#pragma preserve を指定し演算器とすることによって処理の高速化を行う。

(E) 画像分割による並列化

水平に画像を分割してそれぞれ別の外部メモリに格納し、分割の数だけ相関値計算回路を用意することで並列処理を行う。画像を分割する際に生じる境界部分においてはオーバーラップ部（重複）分を設けるこ

とによって精度を保つ。このときマスク画像の中心と最下位の行との距離が3ピクセルなためオーバーラップは3行分設けなければならない。

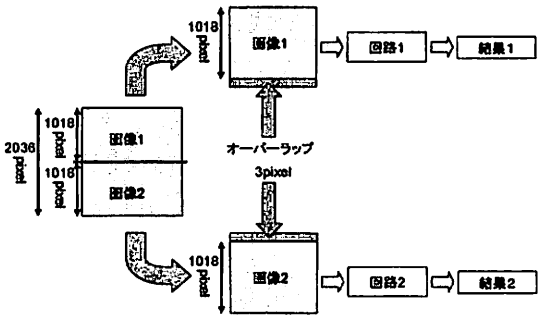


図 3.5 画像分割による並列化

4. 実装結果と考察

3章で述べた各高速化手法を回路に実装し、実際の入力画像サイズの 1008×1018 画素の画像を用いて処理速度と回路面積を求めた。

処理時間は Model Sim によるシミュレーションを行った結果であり、回路面積は Design Compiler により合成した結果である。回路の動作周波数は 100MHz、ライブラリは HITACHI0.18μm を用いた。

作成した回路は、以下(ア)−(キ)である。

(ア) 逐次処理
(イ) 相関値判別式の変形＋データ再利用によるメモリアクセス数の削減
(ウ) (イ)＋相関値判別式の演算回路化
(エ) (ウ)＋二次微分による粒子判別
(オ) (エ)＋レジスタ追加によるメモリアクセスの削減
(カ) (エ)＋画像分割による並列化（2並列）
(キ) (エ)＋画像分割による並列化（7並列）

回路(イ)では相関値判別式を変形することによって逐次処理回路（ア）より約 34,000 ゲートの増加であったが 820[ms]まで高速化でき、約49倍の高速化を実現した。相関値判別式の演算回路化、二次微分による粒子判別により 372[ms]まで高速化した。

また、この回路を7並列化した回路（キ）で 61[ms]となり目標の 66.66[ms]以内で処理を行える回路を作成することができた。この回路は元の逐次処理回路（ア）と比較すると回路規模は 12 倍であるが、654倍の処理速度の向上となった。

表 4.1 ハードウェアの高速化手法実装結果

	処理時間 (ms)	回路規模 (ゲート)
ソフトウェア	21,780	-
(ア)	39,913	59,097
(イ)	820	93,456
(ウ)	450	109,667
(エ)	372	98,734
(オ)	228	106,277
(カ)	207	217,697
(キ)	61	799,227

文献[6][8][9]で提案した回路では、分割された外部メモリを追加使用し、同時にアクセスを行うことでメモリアクセス回数を削減し、かつ 1pixel キャッシュ処理による高速化手法を用いて、処理時間を 2,550[ms]まで高速化していた。しかしこの手法は外部メモリを使用しなければならない。今回実装を行ったデータ再利用によるメモリアクセス数の削減は、入力画像を格納する外部メモリのみで実現できる。また、相関値判別式の変形は1ピクセル処理をより効率的に行っている。

5. 粒子追跡システムの現状と今後の課題

現在設計している粒子追跡システムの各処理時間を表 5.1 に示す。

表 5.1 粒子追跡システムの各処理時間

	処理時間 (ms)	回路規模 (ゲート)
Imread	設計中	設計中
Cor	61	799,227
Chain	264	173,369
KC	107	1,657,515

現状の粒子追跡システム回路では今回最適化を行った Cor 関数で目標処理時間の 66.66[ms]以内を実現することができた。しかし、Chain 関数がボトルネックとなるため、現在は 264[ms]単位でのパイプライン処理となり、リアルタイムはまだ実現できていない。今後の課題として、Chain 関数の高速化を行うとともに、KC 関数についても同様に高速化と回路規模削減を行っていかなければならない。

また、Imread 関数については加算とビットシフトのみで行うことができる処理である。Imread 関数後の処理である Cor 関数では相関値計算を行う回路にまだ少し時間に余裕があるため、この回路内に Imread 関数の処理を追加してやることによって Cor 関数の処理時間

を増加させることなく同一回路内で処理を行えるようにする。このようにすることによって全体の構造を66.66[ms]で動作するCor, Chain, KCの3つのモジュール構成にすることができ、3段パイプラインにすることによって1枚の画像の処理時間を266.64[ms]から199.98[ms]で処理できるようになる。

なお、元のソフトウェアの処理時間は27[s]であり、このシステムと比較すると約103倍の高速化を実現している。

6. まとめ

本研究では、粒子追跡システムのリアルタイム化を行うにあたり、相関値計算部の処理のハードウェア化を中心に行い、5種類の高速化手法を実装し、目標の処理時間を達成することができた。

謝辞

粒子追跡技術を使用した研究を行うにあたり、粒子追跡法のソースとして粒子マスク相関法を提供し、ソフトウェアの解析にあたって直接ご指導いただいた近畿大学理工学部社会環境工学科江藤剛治教授、竹原幸生助教授に深くお礼申し上げます。

Bachを用いたハードウェア設計を実現するに当たり、多大なるご指導を頂いたシャープ株式会社山田晃久様をはじめ、Bach開発グループの皆様に厚くお礼申し上げます。

また、本研究の一部は、東京大学大規模集積システム設計教育研究センターを通し、Synopsys株式会社のツールを用いて行われたものである。

文　献

[1] K. Okada, A. Yamada, T. Kambe; "Hardware Algorithm Optimization Using Bach C," IEICE Trans. Fundamentals vol.E85-A, No.4, (pp835-841), 2002.

[2] 可視化情報学会編：“PIVハンドブック”、森北出版株式会社、東京、1～4（2002）

[3] 江藤剛治、竹原幸生、道奥康治、久野悟志：“PTVのための粒子画像抽出法に関する検討-粒子マスク相関法について-”、水工学論文集、第40巻、1996年2月

[4] 江藤剛治、竹原幸生、岡本孝司：“標準画像を用いた粒子マスク相関法とKC法の性能評価”、日本機械学会論文集、65、184～191（1999）

[5] 栗原孝次：“データの科学”、放送大学教育振興会、2001年3月

[6] 大口貴裕、大倉崇宜、小西徹也、上甲憲市、神戸尚志：“粒子追跡技術のハードウェア化の一手法-相関値計算部のハードウェア化-”電子情報通信学会、信学技報 CPSY2004-104、pp85～90、2005年3月

[7] 上甲憲市、大口貴裕、小西徹也、大倉崇宜、神戸尚志：“粒子追跡における KC 法のハードウェア化” 電子情報通信学会、信学技報 VLD 2005-1、pp1～6、2005年5月

[8] 上甲憲市、大口貴裕、上津寛和、酒井皓司、大倉崇宜、神戸尚志：“C言語設計によるリアルタイム粒子追跡システム” 電子情報通信学会、信学技報 VLD 2005-176、pp31～36、2005年12月

[9] 上津寛和、大口貴裕、上甲憲市、酒井皓司、神戸尚志：“C言語による粒子追跡システムの設計” 電子情報通信学会、信学技報 VLD-2005-119、pp67～72、2006年3月