実時間分散画像処理システムのための信頼度駆動アーキテクチャ

吉本, 廣雅
九州大学システム情報科学研究院知能システム学部門

有田, 大作
九州大学システム情報科学研究院知能システム学部門

谷口, 倫一郎
九州大学システム情報科学研究院知能システム学部門

出版情報：情報処理学会研究報告. 計算機アーキテクチャ研究会報告. 2004 (20), pp.85-90, 2004-03-01. 情報処理学会
バージョン：

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

2004－ARC－157 (15)
2004－HPC－ 97 (15)
2004／3／2

実時間分散画像処理システムのための信頼度駆動アーキテクチャ

吉本廣雅[†] 有田大作[††] 谷口倫一郎[††]

動画像という莫大な帯域を持つストリームを逐次解析するアプリケーションでは，出来るだけ高精度で高速な処理が要求される．しかし利用可能な計算機資源には限りがあり，システム設計者は利用可能な計算機資源の下で処理時間と精度のトレードオフを考えなければならない．つまり遅延は小さいが精度は悪いアルゴリズム，遅延は大きいが精度はよいアルゴリズムを状況に応じて使い分ける必要がある．

信頼度駆動アーキテクチャとは，事前知識として与えられた精度と遅延の関係式を用いて，状況に応じて最適なアルゴリズムを選択する機構である．本稿では信頼度駆動を用いた分散共有メモリ機構を実現し，動画像から人間の動作を実時間で推定するアプリケーションを例にその有効性を示す．

Confidence-driven Architecture for Real-time Distributed Image Processing System

Hiromasa Yoshimoto,[†] Daisaku Arita[††] and Rin-ichiro Taniguchi[††]

The algorithm for vision-based processing has a trade-off between the precision and the delay. The more complex algorithm becomes, the bigger delay will be. On the other hand, the delay should be small to keep throughput higher. So we have to choose the best algorithm which suits the situation.

In this paper, we deal this trade-off as the synchronization matter of the confidence. We implement those algorithms as threads, and execute one as soon as satisfied with the its confidence, so that we can choose the best algorithm dynamically.

1. はじめに

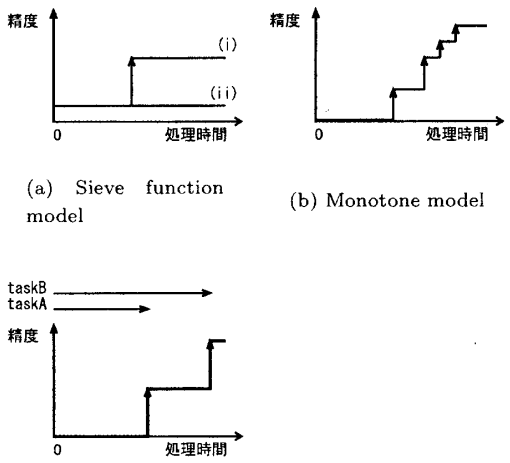
動画像を処理し何らかの情報を取り出すアルゴリズムには精度と遅延のトレードオフの問題がある．出力の精度を向上させるためには，一般に複雑なアルゴリズムを適用する必要があり，処理時間は長くなる．一方で，実時間性が要求されるアプリケーションではある入力に対応する出力が得られるまでの遅延も考慮しなければならない．すなわち，出力の品質を高めるためには，精度と遅延のトレードオフを考慮しつつ状況に応じてアルゴリズムを切り替える必要がある．

精度と遅延のトレードオフの関係は動画像のようなメディアデータを扱う実時間システムでは広く一般に認識されている．たとえばLiuらは実時間性を保つために不完全な(Impreciseな)計算結果を利用する手法を計算時間に応じて出力の精度が向上するImprecise計算モデルとして定式化している[1)]．しかし，Imprecise計算モデルを実装するための汎用的なソフトウェア構成法はまだ十分に確立されていない．これは，品質を判断することができるのは人間だけであり，精度と遅延の関係は経験的にしか得られない点に原因がある．

信頼度駆動アーキテクチャは，精度と遅延に関する経験的に得られた知識を利用してアルゴリズムの動的な選択を行うための機構である．アルゴリズムの精度を相対的に比較するための尺度として信頼度を導入し，精度と遅延の関係に関する人間の知識を信頼度の時間関数として定義できるようにする．信頼動駆動アーキテクチャでは，信頼度の時間関数として与えられた知識を元にImprecise計算モデルを生産者消費者間の同期の問題として取り扱う．

本稿では，信頼度駆動アーキテクチャの概要と，信頼度駆動を用いた分散共有メモリ機構について述べる．動画像から人間の動作を実時間で推定するアプリケーションを例に，Imprecise計算モデルがスレッド間の信頼度による同期の問題として取り扱え，精度と遅延

† 九州大学 大学院システム情報科学府
Department of Intellient Systems, Kyushu University
†† 九州大学 大学院システム情報科学研究院
Department of Intellient Systems, Kyushu University



(a) Sieve function model

(b) Monotone model

(c) Multiple versions model

図1 Imprecise 計算モデルの概要

のトレードオフが容易に制御できることを示す.

2. 研究背景と関連研究

2.1 Imprecise 計算モデル

精度と遅延のトレードオフの問題は，スケジューリングの問題である．利用可能な計算機資源の下で最善の出力を得るためには，利用可能な計算時間や通信路の帯域を考慮し，システムを構成する各タスクを最適な順序で動作させる必要がある．また，様々な計算機環境でつねに最善の出力を得るためには実行時の動的なスケジューリングが必要となる．

Liuらは実時間システムにおけるスケジューリングの問題を解決するためにインプリサイス計算モデルという計算モデルを提案した[1]．一般的な計算モデルでは，完全に計算が完了したタスクの計算結果のみが計算結果と見なされ，処理を中断させたタスクの計算結果は利用できない．しかしインプリサイス計算モデルは，計算途中で生成された不正確(Imprecise)な結果を利用できる計算モデルである．計算を進めると進み具合いに応じてその計算結果の品質が高まると考えことで，Imprecise計算モデルにそって記述されたプログラムは計算タスクの途中終了が可能となる．その結果，スケジューリングに関して大きな柔軟性を持つようになる．

このImprecise計算モデルには，3つの計算モデルがある．**図1**にそれぞれの計算モデルでの，計算時間（横軸）と計算結果の精度（縦軸）の関係を示す．

図1(a)は Sieve function model と呼ばれる計算モデルである．これは，処理時間に余裕がなければ処理そのものを省略し，従来の情報を基に推定した結果で代用するタスクモデルである．たとえば，30fpsで獲得された動画像をリアルタイムに遠隔地に転送し30fpsで表示するアプリケーションを考える．つねに30fpsの表示を行うためには，転送が間に合わない場合は前回受信したフレームをそのまま表示するといった処理が必要となる．

2つ目の計算モデルは，Monotone model と呼ばれる．図1(b)に示されるように，このモデルは計算時間に比例して処理結果の品質が向上する計算モデルである．たとえば探索のアルゴリズムを考えると，探索空間を探索すればするほど探索結果の精度は高くなる．

3つ目の計算モデルは，Multiple versions model と呼ばれる．図1(c)に示されるように，このモデルでは，2つのタスクを並列に起動する．図中の$TaskA$は精度が悪いが処理時間は短いアルゴリズムによるタスクである．$TaskB$は対照的に，精度が良いが処理時間が長いアルゴリズムによるタスクである．精度を犠牲にしてでも遅延の少ない処理結果が必要な場合は$TaskA$の結果を利用し，高い精度が必要な場合は，$TaskB$の結果を利用することができる．

このように，Imprecise計算モデルは広く一般に使われている様々なソフトウェア構成手法をモデル化したものである．しかしながら，その汎用的な実装技術はいまだに確立されていない．

2.2 推定器の利用

通信を伴う分散システムでは通信により生じる遅延の削減が重要である．さらに通信路には遅延の揺らぎが存在する．遅延やその揺らぎは，分散システムの性能に大きな影響を与える．

高速なネットワークが用意できればデータの転送により生じる遅延を削減することができる．しかし遅延やその揺らぎをゼロにすることは不可能である．一方，予測を用いれば遅延やその揺らぎをゼロにできる．予測は，情報を得るために本来必要な送受信処理を省略し代わりに予測された情報で代用する手法と考えると，Imprecise計算モデルにおける Sieve function model とみなせる．

予測はすでに様々なアプリケーションで利用されている．たとえば，ネットワークバーチャルリアリティの分野では推測航法（Dead reckoning）と呼ばれる手法を用いて通信の遅延を減少させる手法が一般的に用いられている[2]．これは，仮想空間上で各ユーザの位置を共有する際に新しい位置情報が受信されるまでは過去の位置情報から現在の位置情報を予測する手法で

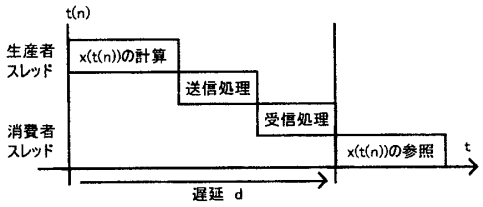


図2 遅延の定義．生産者スレッドの出力は，計算と通信に必要な時間だけ遅れて消費者スレッドへ届く．

ある．推測航法を用いると，経験的に誤差が小さいと考えられる間は通信を省略することが可能であり，通信量や遅延を削減することができる．

また，日浦らは共有メモリ上に記録された離散的時系列データを基に任意時刻の値を補間する関数を利用することで，複数の並列プロセス間でのデータ交換を完全に非同期に行う機構をダイナミックメモリとして定式化している[3]．

推測航法やダイナミックメモリは予測器を介した通信手法である．しかし実際には，1秒前の過去の値と同じ精度で1時間後の遠い未来の値を予測することは不可能である．このように推測航法やダイナミックメモリでは予測により得られる情報の正確さについてはあまり十分に考慮していない．つまり精度より遅延の削減を優先させるため同期を完全に無くした結果，得られる情報の正確さまでもが失われている．

3. 信頼度駆動について

信頼度駆動では，Imprecise計算モデルを生産者消費者スレッド間の通信の問題として取り扱う．

図2に示されるように，ある時刻$t(n)$の入力に対する生産者スレッドの出力$x(t(n))$は，計算と通信に必要な時間だけ遅れて消費者スレッドへ届く．計算と通信に必要な時間を遅延dとすると，消費者スレッドは時刻$t(n)+d$以降でなければ，データ値$x(t(n))$を参照することはできない．そのため，消費者スレッドはデータ値$x(t(n))$が得られるまで待つ必要があり，生産者スレッドと消費者スレッドは同期して動作する．

一方，Imsprecise計算モデルに従うことで計算や通信による遅延を削減できる．たとえば過去に受信したデータ値の履歴hから任意の時刻tのデータ値$x(t)$を推定する関数$F_x(h,t)$が定義できれば，生産者スレッドは消費者スレッドと非同期に動作できる．

図3に示されるように，これらスレッド間の通信方法は精度と遅延のトレードオフの関係にある．非同期通信により消費者スレッドが得たデータ値$x(t)$は，予測の結果得られた値は不正確な値である．Imprecise計算モデルを利用して得られた不正確な結果と，計算

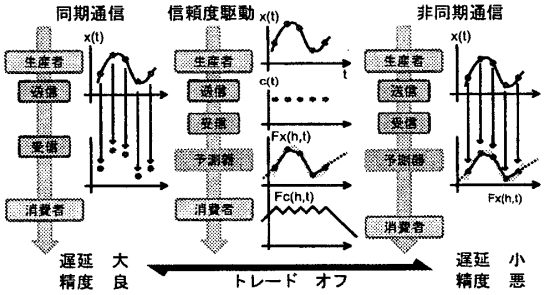


図3 履歴$h=\{t(n),x(t(n)),c(t(n))\}$と予測関数$F_x(h,t)$およびその信頼度関数$F_c(h,t)$の関係

や通信を省略せずに行って得られた本来の出力結果の差，つまり$|x(t)-F_x(h,t)|$の大きさについて議論する必要がある．

しかし，$F_x(h,t)$により予測された結果に含まれる誤差や，アルゴリズムの正確さは，経験的にしか得ることができない．そこで本研究では，アプリケーションとその利用環境を限定した上で，これらの要素に対しての知識を相対的な数値で表現することを考える．そのために信頼度（confidence）という新しい尺度を導入する．信頼度は0から1までの値を持ち，大きな値の信頼度を持つデータ値ほど経験的に品質の高い信頼できる情報であるとみなすことができる．

本研究の中心的なアイデアはこの信頼度という尺度の利用である．図3に示されるように，ある時刻tでの$|x(t)-F_x(h,t)|$の大きさの程度を予測する関数$F_c(h,t)$を経験的に与えることができれば，$F_c(h,t)$の信頼度の大小により，同期と非同期のどちらの通信方法を利用するかを動的に決定できる．すなわち，消費者スレッドが要求する信頼度を**要求信頼度**C_rと定義すると，消費者スレッドは$F_c(h,t)\geq C_r$であれば$F_x(h,t)$による予測されたデータ値を参照し，$F_c(h,t)<C_r$であれば新たに生産者スレッドにより履歴hが更新されるまで待つ．つまり，$F_c(h,t)$とC_rの大小関係による信頼度の同期機構を実現できる．

このように，精度と遅延の関係のような経験的にしか得られない知識を計算機で扱いやすいように信頼度の尺度を用いて数値として表現することで，本来経験的にしか取り扱えなかった精度と遅延の関係を単純な信頼度の大小の比較で取り扱えるようになる．信頼動駆動アーキテクチャとは，このような人間が経験的に獲得した知識を信頼度という形で計算機に与えることで経験的に最善なスケジューリングを行う機構である．

3.1 信頼度駆動メモリ

信頼動駆動メモリとは，信頼度駆動の概念に基づいた共有メモリ機構である．

過去ある時刻$t(n)$での，生産者スレッドの出力とそ

の信頼度の組$\{t(n), x(t(n)), c(t(n))\}$の集合を履歴$h$とすると，任意の時刻$t$の$x(t)$とその信頼度$c(t)$を推定する関数が，それぞれ

$$x(t) = F_x(h, t)$$
$$c(t) = F_c(h, t) \qquad 0 \leq c(t) \leq 1$$

と定義できるとする．

生産者消費者の関係にあるスレッドは履歴を共有メモリ上で共有することで，以下の操作により同期非同期を切り替えながら任意の時刻のデータ値$x(t)$を共有することが可能となる．

- *cdm_write*(m, t, x, c)操作
  信頼度駆動メモリmに時刻tの値を書き込む．
  具体的には，信頼度駆動メモリmに対応付けられた履歴hに，$\{t, x(t), c(t)\}$の組みを追加し，信頼度の同期待ちのスレッドを起床させる．
- *cdm_poll*$(mems, N, M, deadline)$操作
  信頼度による同期を取る．
  具体的には，N個の信頼度駆動メモリに対する要求信頼度をcdm_poll_t 型(**図4**参照)の配列$mems$で指定することで，N個の信頼度駆動メモリのうちM個$(M \leq N)$の信頼度駆動メモリの信頼度が十分になるか，時刻$deadline$が経過するまで，スレッドを休眠させる．
- *cdm_read*$(m, t, C_r, deadline)$操作
  信頼度駆動メモリmから，任意の時刻tのデータ値$x(t)$とその信頼度$c(t)$の予測値の組を読み出す．また0以上の要求信頼度C_rが指定された場合は，内部で前述のpoll 操作を用いることで，信頼度による同期を取る．つまり，得られる信頼度$c(t)$が要求信頼度C_rを満たさない場合は，$deadline$により指定された時刻まで read操作は 休眠 する．信頼度による同期が取れず$deadline$を超過した場合は， その時点での$\{x(t), c(t)\}$ を返す．
- *cdm_open*$(name)$操作
  $name$という名前の信頼度駆動メモリを生成する．
- *cdm_close*(m)操作
  信頼度駆動メモリcdmを解放する．
- *cdm_set*(m, sz, F_x, F_c)操作
  信頼度駆動メモリmに，データ値のサイズszと，データ値の予測関数F_xと，その信頼度の予測関数F_cを設定する．
  F_xとF_c は C言語で記述された関数へのポインタである．ユーザは，履歴hから任意の時刻tの値を推定する関数をC言語で記述し，当set 操作によりシステムに登録する必要がある．

図6，**図5**に，*cdm_write*操作，*cdm_read*操作の処理の概要を示す．*cdm_write*操作により履歴が更新されない限り信頼度は上昇しないため，*cdm_read*操作は十分な信頼度が得られない場合，*cdm_write*操作が発行されるまでスレッドを休眠させる．

```
typedef struct{
   cdm_t    cdm;        /* CDMへのハンドル */
   timespec time;       /* 時刻 */
   conf_t   required;   /* 要求信頼度 */
   conf_t   confidence; /* 得られた信頼度*/
} cdm_poll_t;
```

図4 cdm_poll_t構造体．cdm_pollで使用される．

```
1 cdm_read(m, t, Cr, deadline)
2 {
3   while ( Fc(m.h, t) ≤ Cr ){
4     ret = cond_timedwait(m.cond, deadline);
5     if ( TIMEOUT == ret ) break;
6   }
7   return {Fx(m.h, t), Fc(m.h, t)};
8 }
```

図5 cdm_read操作

```
1 cdm_write(m, t, x, c)
2 {
3   insert(m.h, {t, x, c});
4   cond_signal(m.cond);
5 }
```

図6 cdm_write操作

3.2 分散環境への対応

CDMを分散共有メモリへと拡張するには，*write*操作により履歴に追加される値$\{t(n), x(t(n)), c(t(n))\}$を他のノードに転送するだけで良い．重要な点は，ノード間で履歴が厳密な一貫性を保つ必要が無い点である．履歴は信頼度が要求信頼度を上回る程度におおよそ共有できればよいため，ノード間の通信は厳密なトランザクションを必要としない．単に，*write*操作により履歴に追加される値 $\{t(n), x(t(n)), c(t(n))\}$を他のノードに転送するだけで良い．

4. 実　　験

分散信頼度駆動メモリを用いて，動画像から人間の動作を推定するマーカレスモーションキャプチャシステムを構築した．これは，**図7**(a) のような画像から，対象人物の姿勢を推定し，図7(b)のようなCGを出力するアプリケーションである．実時間で画像から人間の姿勢を推定することができれば，たとえば不審者を自動で検出する防犯システムなどへの応用が期待される．

実験では，マーカレスモーションキャプチャシステムを分散信頼度駆動メモリを用いてPCクラスタ上に実装した．概要を**図8**に示す．実験環境としては，Pentium4 3.2GHzのCPUを搭載したRedhat9ベースの

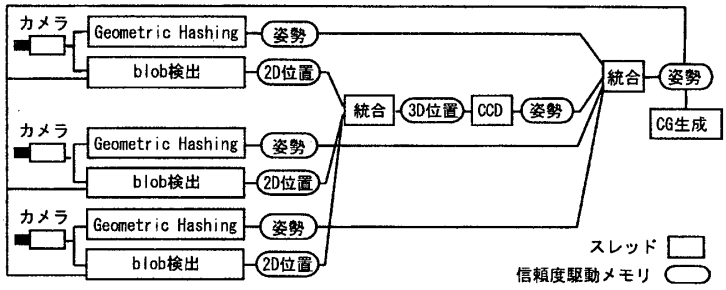


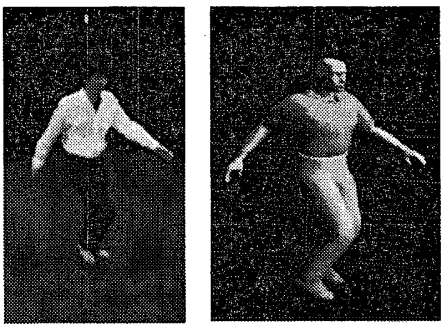
図8 マーカレスモーションキャプチャシステムの概要

PC10台により構成されるPCクラスタを利用した．PC間の接続は100Base-TXであり，UDPを用いた．このPCクラスタ上で，9台のカメラから獲得された動画像を22個のスレッドで処理した．

人間の体を多関節物体としてモデル化すると，肘や膝の関節の回転角度により人間の姿勢は一意に決定できる．つまり，カメラにより獲得された画像から人間の姿勢を推定するアルゴリズムは各関節の回転角度を推定するアルゴリズムである．今回の実験では，二つのアルゴリズムを用意し，信頼度駆動によるアルゴリズムの動的な切り替えを行った．以下，二つのアルゴリズムの概要を説明する．

4.1 Geometric Hashing[4] を用いた姿勢推定

前述のように人間の姿勢は関節の回転角度により一意に決定できる．そこで事前に関節の回転角度と画像上の特徴点の関係を表として計算しておけば，入力画像の特徴点から表を探索することで対応する姿勢を求めることができる．ハッシュ表を用いて探索空間を絞り込む手法である Geometric Hashing[4] を応用し，手と胴体などがあまり重ならない判別しやすい特定の姿勢のみから表を作ることで，500$msec$程度の処理時間で特定の姿勢が推定できる．



(a) 入力画像例　　(b) 出力画像例

図7 アプリケーション概要

4.2 CCD法[5] を用いた姿勢推定

手や顔の領域(blob)は肌色である事など画像上の特徴を経験に基づいて処理すると，画像上での手や顔や足の2次元位置を推定することができる．複数の視点の画像から得られたblobの2次元位置とカメラ間の幾何学的関係を用いることでblobの3次元位置が計算できる．

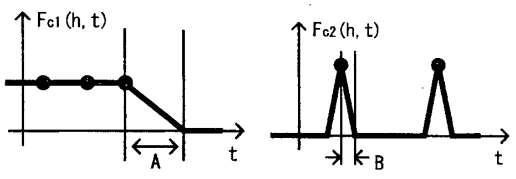
blobの3次元位置から各関節の回転角を求める際にはCCD法を用いた．CCD法とは，式(1)を最小にする関節角$\theta_1 \ldots \theta_N$ を反復により発見的に求める手法の一つである[5]．

$$\sum_{i=1}^{N} |\vec{\mathrm{P_i}}(\theta_1 \ldots \theta_N) - \vec{\mathrm{G_i}}| \qquad (1)$$

ここで $\vec{\mathrm{P_i}}(\theta_1 \ldots \theta_N)$ は，各関節角$(\theta_1 \ldots \theta_N)$に対応する体の各部位(両手，両足，頭部)の3次元位置を返す関数であり，G_iは画像から得られた各部位の3次元位置である．

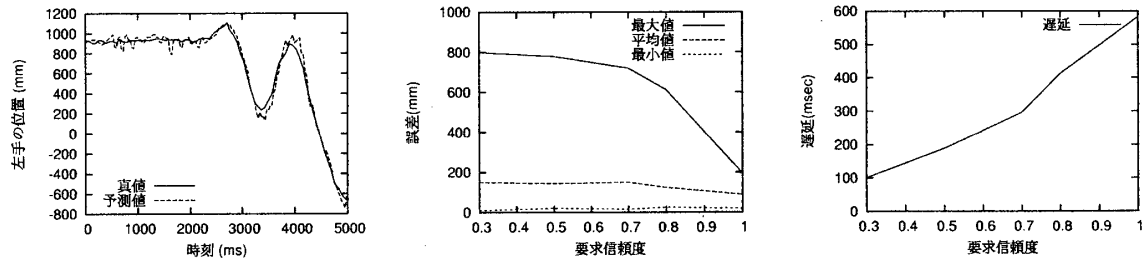
4.3 信頼度駆動の導入

Geometric Hashingを用いたアルゴリズムは精度を優先したもの，CCD法を用いたアルゴリズムは遅延の削減を優先したものである．そこで，図8に示されるようにこれらアルゴリズムをスレッドに分割し信頼度駆動メモリを介した通信を行うことで，信頼度駆動を行った．信頼度駆動メモリを使う場合には，$F_x(h,t)$と$F_c(h,t)$の二つの推定関数を知識として与える必要



(a) $F_{c1}(h,t)$の定義　　(b) $F_{c2}(h,t)$の定義

図9 信頼度の予測関数$F_{c1}(h,t)$，$F_{c2}(h,t)$の定義



(a) 左手の実際のx座標と予測されたx座標

(b) 要求信頼度と誤差

(c) 要求信頼度と遅延

図10 手の動きと誤差の関係．及び，要求信頼 度と誤差，遅延の関係

がある．

図8中の2D位置，3D位置を保持する信頼度駆動メモリには$F_x(h,t)$として線形予測する関数と，信頼度の関数として**図9**(a)を与えた．

一方，姿勢を保持する信頼度駆動メモリには$F_x(h,t)$としては関節角を線形予測する関数を与えた．信頼度の関数は，Geometric Hashingを用いたアルゴリズムの結果を保持する信頼度駆動メモリには図9(b)を，それ以外の姿勢を保持する信頼度駆動メモリには図9(a)を与えた．

4.4 性能評価

手の実際の位置と予測された手の位置の例を**図10**(a)に，要求信頼度と誤差，遅延の関係をそれぞれ図10(b)，(c)に示す．

図10(a)に示されるように予測結果と実際の位置の差は当然，対象物の動作速度の影響を受ける．しかし，図10(b)，(c)に示されるように，要求信頼度を高くすると誤差は小さくなるものの遅延は大きくなり，逆に要求信頼度を小さくすると誤差は大きくなるものの遅延は小さくなることが観測できた．このように制御と遅延のトレードオフの問題に対して，要求信頼度という尺度でどちらを優先するのかを選択できる機構が実現できた．

5. おわりに

多種多用の現実世界を堅牢に取り扱える画像処理システムを作る上では，様々な性質の異なるアルゴリズムを知的に統合する機構は必要不可欠である．今回の実験では，信頼度駆動機構を利用することで分散環境下において信頼度の尺度で人間が与えた事前知識を元に精度と遅延のトレードオフを柔軟に制御できるアプリケーションが実現できることを確認した．

今後の課題としては，複数の生産者が同時に起床した場合に信頼度に応じてどのスレッドを優先させるべきかを決定するスケジューラの開発が急務である．莫大な数の画像処理アルゴリズムをスレッドとして実装し，信頼度により同期が取れたスレッドを逐次実行することで，時々刻々と変化する現実世界に追従できる知的な画像処理システムの実現を目指している．

参考文献

1) W.S.Liu, J., Shih, W., Lin, K., Bettati, R. and Chung, J.: Imprecise Computations, *Proceedings of the IEEE*, Vol. 82, No. 1, pp. 83–94 (1994).
2) Singhal, S. and Zyda, M.: *Networked Virtual Environments*, Addison Wesley (1999).
3) 日浦慎作, 村瀬健太郎, 松山隆司: ダイナミックメモリを用いた実時間対象追跡, 情報処理学会論文誌, Vol. 41, No. 11, pp. 3082–3091 (2000).
4) Wolfson, H. J. and Rigoutsos, I.: Geometric Hashing: An Overview, *IEEE Computational Science & Engineering*, Vol. 4, No. 4, pp. 10–21 (1997).
5) Wang, L.-C. and Chen, C.: A combined optimization method for solving the inverse kinematics problem of mechanical manipulators, *IEEE Transactions on Robotics and Automation*, Vol. 7, No. 4, pp. 489–499 (1991).