実時間ビジョンシステムのための信頼度駆動メモリ

吉本廣雅[†] 有田大作[††] 谷口倫一郎[††]

画像から精度の高い情報を得るためには，莫大な計算量が要求される．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は小さければ小さいほどよい．つまり，計算結果の精度と遅延のトレードオフを考慮する必要がある．筆者らは，信頼度駆動という新しい概念により，精度と遅延のトレードオフの問題を簡潔なモデルで定式化する．そして，このモデルを利用するためのプログラミング・インタフェースとして信頼度駆動メモリを実装する．本論文では，シミュレーションおよびアプリケーションによる実験の結果から，信頼度駆動メモリの有効性を示す．

Confidence-driven Memory for Vision System

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

To get accurate information from input-images, the huge amount of computation power is required. On the other hand, in real-time system we must take care of the latency. Under given hardware resources, we must make difficult trade-off between the precision and the latency. In this paper we propose Confidence-driven scheme and formulize the problem as a simple model. Here, we will present the Confidence-driven Memory architecture as its programming interface and show its efficiency based on some experimental results.

1. はじめに

コンピュータビジョンの技術を利用することで，我々は画像から物体の位置，姿勢などの情報を得ることができる．これらの情報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，精度と遅延のトレードオフが存在し，アプリケーションに応じてどちらかを犠牲にするソフトウェア構成法が必要である．

このようなソフトウェア構成法の概念は，すでにImprecise計算モデルとして定式化されている[1)]．このモデルは，計算時間に応じて出力の精度が向上する計算モデルである．計算途中の結果も利用できる特性を利用することで，精度と遅延のトレードオフの問題を容易に取り扱うことができる．

しかしながら，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．我々は，この問題を解決するために信頼度駆動メモリを提案する．信頼度駆動メモリは，過去の情報の履歴を参照して任意の時刻の情報を推定する推定器と，信頼度による同期機構から構成される．信頼度駆動メモリを用いることで，Imprecise計算モデルは生産者消費者の関係にある2つのタスク間の同期の問題に置き換えることが可能である．

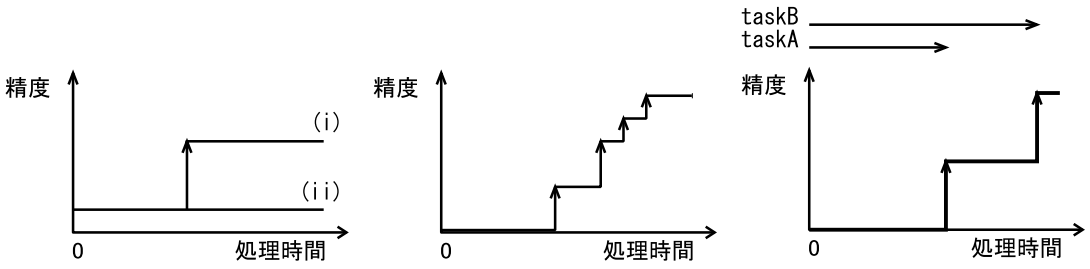
以下，本論文では，まず2章で関連研究と研究の背景を述べる．続いて3章で信頼度の概念と信頼度駆動メモリの構造を示し，Imprecise計算モデルとの関係を示す．最後にシミュレーションと実アプリケーションによる実験の結果を示し，信頼度駆動メモリを利用することで，精度と遅延のトレードオフの問題を容易に制御できることを示す．

2. 研究の背景と関連研究

2.1 Imprecise計算モデルの利用

精度と遅延のトレードオフの問題は，スケジューリ

† 九州大学大学院システム情報科学府
Graduate School of Information Science and Electrical Engineering, Kyushu University
†† 九州大学大学院システム情報科学研究院
Faculty of Information Science and Electrical Engineering, Kyushu University



(a) Sieve function model (b) Monotone model (c) Multiple versions model

図 1 Imprecise 計算モデルの概要
Fig. 1 Overview of Imprecise computation model.

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新しい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）な結果も利用できる計算モデルである．計算の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る 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，30 fps で獲得された動画像をリアルタイムに遠隔地に転送し 30 fps で表示するアプリケーションを考える．つねに 30 fps 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．たとえばテンプレートマッチングにより画像から対象物を探索するアルゴリズムを考えると，できるだけ多くのテンプレート画像とのマッチングをとるほうが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

コミュニケーションの媒体となるようなアプリケーションでは，遅延の削減が重要である．特に遠隔地の他者とコミュニケーションをとるような場合，通信における遅延の削減の問題を考えなければならない．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，遠隔地の他者と共有する情報の品質を大きく下げる要因となる．

高速なネットワークが用意できれば高速にデータを転送することは可能である．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，予測を利用することでこれらの問題を隠蔽することは十分に可能である．予測は，情報を得るために必要な処理そのものを省略し，代わりに予測された情報で代用する手法と考えることができる．必要に応じて予測を利用する

手法は，Sieve function model そのものである．

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，たとえば仮想空間上で各ユーザの位置を共有する場合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推定する手法である．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推定器を介した通信手法ととらえることができる．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，デッドレコニングやダイナミックメモリでは推定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．つまり同期を完全になくした結果，得られる情報の正確さまでもが失われている．

3. 信頼度駆動メモリの概要

3.1 遅　　延

まず本論文では遅延を，時刻 $t(n)$ から情報 $x(t(n))$ が参照可能になるまでの時間，と定義する．例として，生産者と消費者の関係にある2つのタスク T_p，T_c を考える．**図2** (a) に示されるように，生産者の立場にあるタスク T_p は，時刻 $t(n)$ に獲得された n 番目の画像を入力とし，その画像から取り出した情報 $x(t(n))$ を出力する．この情報 $x(t(n))$ は消費者の立場にあるタスク T_c により利用される．そのため，タスク T_p とタスク T_c は，$\{t(n), x(t(n))\}$ の組を通信し情報の同期をとる必要がある．この情報の同期に必要となる時間が遅延である．

遅延時間を d とすると，消費者タスクは $x(t(n))$ を参照するためには同期がとれる時刻 $t(n)+d$ まで待つ必要がある．スループットを向上させるにはこの待ち時間を短くする必要がある．待ち時間を削減するには，

- すでに同期のとれた情報を参照する．たとえば情報 $x(t(n))$ を時刻 $t(n)+d$ 以降に参照するなどして，過去の情報を参照すると遅延はゼロとなる．
- 遅延を削減する．

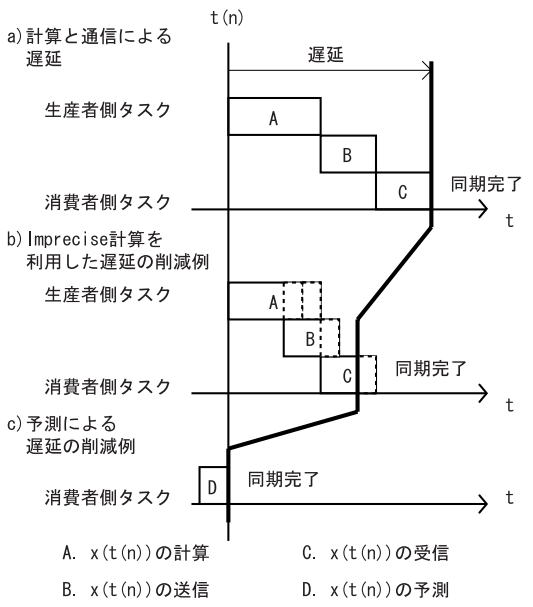


図2 遅延とその削減の例．精度を犠牲にすることで，遅延を削減することができる

Fig. 2 Definition of the latency and its reduction strategy.

のどちらかの方法をとる必要がある．

遅延は，Imprecise 計算モデルを利用し，精度を犠牲にすることで削減可能である．Monotone model を利用した遅延の削減例と，Seive function model を利用した予測による遅延の削減例を図2 (b)，(c) に示す．

計算と通信により生じる遅延は，たとえば図2 (b) に示されるように計算や通信の処理に Monotone model を適用することで削減することが可能である．また，精度をまったく気にしなければ，図2 (c) のように予測により得られた未来の値を利用することで，遅延をゼロにする，つまり時刻 $t(n)$ になると同時に $x(t(n))$ を参照することさえ可能となる．もちろん，これらの手法は精度を犠牲にして遅延を削減する手法であり，実際には精度と遅延のトレードオフが存在する．

3.2 信　頼　度

前節で述べたように，Imsprecise 計算モデルを利用することで，計算や通信による遅延は削減できる．しかしながら，Imprecise 計算モデルや予測の利用により得られた値は，不正確な値である．予測や Imprecise 計算モデルを利用して求めた結果と，計算や通信を省略せずに行って得られた結果の差について議論する必要がある．

予測や推定により得られた結果に含まれる誤差や，Imprecise 計算により得られた結果の精度，アルゴリズムの正確さは，客観的には比較することができない．

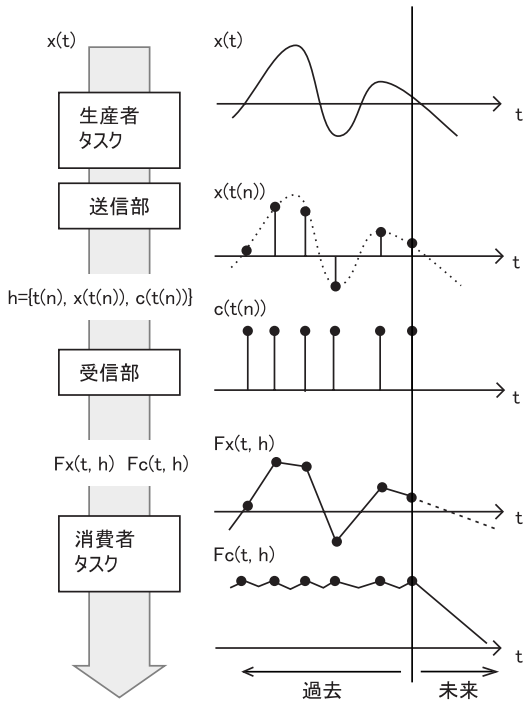


図 3 履歴 $h = \{t(n), x(t(n)), c(t(n))\}$ と予測関数 $F_x(h,t)$ およびその信頼度関数 $F_c(h,t)$ の関係

Fig. 3 Relation among history $h = \{t(n), x(t(n)), c(t(n))\}$, $c(t(n))$, estimation functions $F_x(h,t)$ and $F_c(h,t)$.

しかし，アプリケーションが限定されれば，主観的にこれらの要素を相対的な数値で表現することは可能である．本研究では，この相対的に表現された値を信頼度（confidence）と定義する．信頼度は 0 から 1 までの値を持ち，大きい信頼度を持つ情報ほど品質の高い信頼できる情報であると見なす．もちろん実際の品質と信頼度の値の関連はアプリケーションに依存するものである．したがって，アプリケーションの特性に合わせて，各アルゴリズムの特性や取り扱う対象に関する事前知識などを総合し信頼度は決定されなければならない．

本研究の中心的なアイデアは，この信頼度という測度の利用である．つまり，大きな信頼度を持つ情報ほど品質の向上に貢献する重要な情報であると考えると，スケジューリングによる優先順位の決定は単純な信頼度の大小の比較の問題に置き換えることが可能である．次節では，この考えに基づいた信頼度駆動の概念を説明する．

3.3 信頼度駆動

3.1 節で述べたように，消費者のタスク T_c がより遅延の少ない情報を参照するためには，Imprecise 計算モデルを利用する必要がある．つまり，消費者のタスク

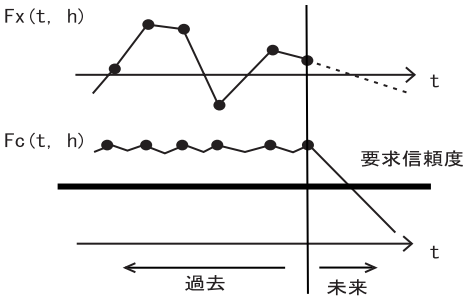


図 4 信頼度駆動の概要．信頼度関数と要求信頼度から十分な信頼度が得られる時間が決定される

Fig. 4 Overview of confidence-driven scheme. Accroding to $F_c(h,t)$ and required confidence, we can decide time when a reliable estimated value will be acquired.

が情報に要求する信頼度を下げることで，インプリサイスな結果を利用する必要がある．一方，T_c が十分な信頼度の情報を要求する場合は，不完全な（Imprecise な）結果は利用することができず，遅延は大きくなってしまう．

このように，消費者側のタスクが要求する信頼度により遅延は増減する．いい換えると，要求する信頼度を制御できれば遅延も制御可能である．本研究では，消費者側のタスクが情報に要求する信頼度を「要求信頼度」と呼ぶ．消費者側のタスクは，要求信頼度を十分満たす情報が得られて初めて処理を開始することが可能である．

本研究では，このように，必要とする情報の信頼度によりタスクが起動されるという考えを信頼度駆動と定義する．つまり，消費者側のタスクがある時刻 t の情報 $x(t)$ を参照する場合，生産者側のタスクの処理結果の信頼度 $c(t)$ が要求信頼度 C_r より大きくなるまでは，消費者タスクは停止する．

3.4 信頼度駆動メモリ

生産者タスクは，ある時刻 $t(n)$ に獲得された画像を入力とし，情報 $x(t(n))$ とその信頼度 $c(t(n))$ を出力する．過去のこれら $\{t(n), x(t(n)), c(t(n))\}$ の値から予測や補完を行うことで，任意の時刻 t の値 $x(t)$ とその信頼度 $c(t)$ を推定することが可能である（**図 3**）．

過去の $\{t(n), x(t(n)), c(t(n))\}$ の値の集合を履歴 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

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

と定義できるとする．この 2 つの関数と履歴 h により，図 3 に示すように，任意の時刻 t での $x(t)$ と $c(t)$ が推定できる．さらに．**図 4** に示すように，信頼度関

数と $F_c(h,t)$ と要求信頼度 C_r の関係からある時刻 t での推定された情報 $x(t)$ が信頼できるか否かを判断することもできる．

信頼度駆動メモリは，この考えを共有メモリの形で実装したものである．信頼度駆動メモリ m は，タスク間で共有された共有メモリ上にある履歴 $h = \{t(n), x(t(n)), c(t(n)\}$ と 事前に定義された 2 つの予測関数 $F_x(h,t)$ と $F_c(h,t)$ から構成される．この信頼度駆動メモリに対して，各タスクは，3 つの操作，*read*，*write*，*poll* 操作を行うことができる．以下，それぞれの機能について説明する．

- $read(m,t,C_r,deadline)$ 操作
  信頼度駆動メモリ m から，任意の時刻 t で推定される $x(t)$ とその信頼度 $c(t)$ を読み出す操作である．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，最大 *deadline* により指定された時刻まで，信頼度が上昇するのを待つ．*deadline* を超えた場合は，その時点での推定結果 $x(t)$ とその信頼度 $c(t)$ が結果として得られる．
- $write(m,t,x,c)$ 操作
  信頼度駆動メモリ m が持つ履歴 h に，$\{t, x(t), c(t)\}$ の組を追加する．
- *poll* 操作
  複数の信頼度駆動メモリの信頼度が十分に揃うまで待つ操作である．具体的には，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それぞれ十分になるまで待つ．

read 操作で読み取られる値は *write* 操作により書き込まれた値そのものではなく，つねに $F_x(h,t)$ を使って予測された値である点が重要である．つまり，2 つのタスクは生産者と消費者の関係にあるが，生産者が *write* 操作を行う回数よりも消費者が *read* 操作を行う回数が多いこともありうる．このように，信頼度駆動による通信を行うことで，2 つのタスクは生産者と消費者の関係にありながら並列に動作することが可能となる．

また，推定結果の信頼度を向上させるには *write* 操作により十分な情報が履歴に追加される必要がある．そこで，*read* 操作を発行した消費者タスクを要求信頼度を満たす情報が得られるまで停止（block）させることで，利用可能な計算機資源をできるだけ *write* 操作を発行する生産者タスクへ割り当て，より早く信頼度が向上するようにしている．

3.5 信頼度駆動メモリと Imprecise 計算モデルの関係

信頼度駆動メモリは，時刻 t とその時の $\{x(t), c(t)\}$ の組を読み書きするインタフェースを提供する．そのため，時間と信頼度の関係を柔軟に取り扱うことが可能である．つまり，Imprecise 計算モデルを簡潔に記述することが可能である．

Sieve function model は，*read* 操作の *deadline* を利用することで簡潔に実現できる．*read* 操作は *deadline* を超過した場合，信頼度が不十分な結果を返す．消費者タスクは *read* 操作の結果の信頼度を参照することで *deadline* を超過したかどうかを判断することができる．この時消費者タスクが *deadline* を超過した不十分な信頼度の情報もそのまま利用すると，Sieve function model に従った処理が実現される．

Monotone model を実現する場合には，信頼度駆動メモリを共有する生産者と消費者の 2 つのタスクを並列に実行する．生産者タスクは，処理時間に比例して逐次信頼度が上昇するようなプログラムを実行し，その結果を逐次 *write* 操作で信頼度駆動メモリに書き込む．消費者タスクは，*read* 操作を行い，信頼度駆動メモリから要求信頼度を満たす推定値が得られるまで待つだけでよい．

ここで，*read* 操作と *write* 操作の内部処理を**図 5** に示す．*read* 操作の待ちの処理は，一般のオペレーティングシステムが提供する基本的な同期機構により実現される．図 5 に示されるように，信頼度が不十分なため同期がとれない場合は，信頼度が揃うまでそのタスクは “スリープ” する．信頼度が上昇するのは *write* 操作により 履歴に新しいデータの組 $\{t,x,c\}$ が追加される場合だけである．そこで，*write* 操作は履歴を更新後，信頼度が上昇するまで待っているタスクを “ウエイクアップ” させる．以上の機構により，信頼度駆動メモリは CPU 時間を浪費することなく信頼

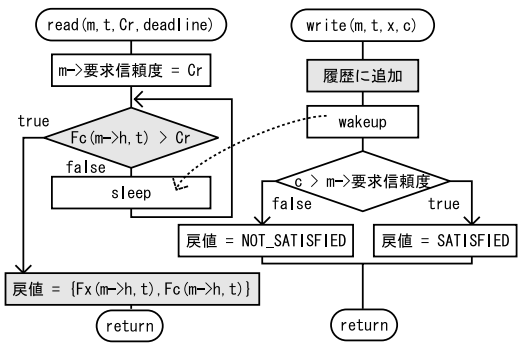


図 5 *read* 操作と *write* 操作の関係．生産者であるタスクが *write* 操作により信頼度駆動メモリに値を書き込み，消費者であるタスクが *read* 操作でその値を信頼度駆動メモリを読み出す
Fig. 5 Relation between a *read* operation and a *write* operation.

度による同期をとることが可能である．

Monotone model を考えると，消費者タスクの *read* 操作が “ウエイクアップ” されるためには，*write* 操作により十分な信頼度を持つ情報が書き込まれる必要がある．一方，一度 *write* 操作が十分な信頼度を書き込めば，それ以上信頼度の高い値を書き込む必要はない．つまり，Multiple versions model に従った生産者タスクは，消費者タスクの要求信頼度より大きい信頼度を *write* 操作により書き込めば，それ以降の処理を打ち切ることができる．十分な信頼度を書き込んだ *write* 操作は $SATISFIED$ という値を返すことで，生産者タスクは *write* 操作の戻り値が $SATISFIED$ になれば処理を打ち切ることができる．$F_c(h,t)$ が定義できるように，消費者タスクのある時刻での要求信頼度は *read* 操作で指定された要求信頼度の履歴から推定可能である．しかしこの推定には，*read* 操作を行うタスクの周期性などを考慮する必要がある．今回の信頼度駆動メモリの実装では，簡単のために，要求信頼度が上昇するとしても急激に上昇することはないという仮定を立て，一番最後の *read* 操作で指定された要求信頼度より少し高い信頼度の情報を書き込めば *write* 操作は $SATISFIED$ を返す．このように，信頼度駆動メモリを利用した Monotone model を実現するためは，2 つのタスクを並列に実行する必要がある．しかし実際には 2 つのタスクは信頼度駆動のもとに並行に実行され，CPU 時間を余計に消費することはない．

最後に，Multiple versioins model を実現する方法を述べる．Multiple versioins model は，複数の信頼度駆動メモリを使用する．つまり，異なるアルゴリズムを実行する N 個の生産者タスクを並列に起動し，その結果が格納された N 個の信頼度駆動メモリから最も信頼できるメモリを選択する．この処理は *poll* 操作により実現できる．*poll* 操作は，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まで待つ操作である．この *poll* 操作により信頼度駆動メモリを選択することは，その信頼度駆動メモリに値を書き込んでいるタスクを選択することと等しい．

このように，信頼度駆動の概念を利用することで，3 つの Imprecise 計算モデルはすべて信頼度の同期の問題に置き換えることができる．つまり，要求信頼度と遅延の 2 つのパラメータを指定することで，精度と遅延のトレードオフの問題として取り扱うことができる．

4. 実　　験

4.1 シミュレーションによる評価

まず，信頼度駆動メモリを介したタスク間通信を行った場合，要求信頼度を変化させることで遅延がどのように変化するかを測定した．実験では，あらかじめ計算機で作成した合成画像から対象物の位置を計算し *write* 操作により信頼度メモリに書き込む生産者タスクと，read 操作により位置を読み出し結果を表示する消費者タスクの 2 つのタスクを用意した．対象物体は半径 R，角速度 ω で等角速度円運度を行う．つまり，時刻 t（単位は ms）での対象物体の位置を $x(t)$ とすると，$x(t) = \{Rcos(\omega t), Rsin(\omega t)\}$ と定義できる．実験では，$R = 193$，$\omega = \pi/180000$ とした．

さらに，信頼度駆動メモリでは 2 つの予測関数が必要である．$F_x(h,t)$ は過去の値は線形補完により，未来の値は線形予測により推定する関数とした．$F_c(h,t)$ は以下のように定義した．

$$F_c(h,t) = \begin{cases} 1 & t < T \\ (1 - \frac{|t-T|}{300})c(T) & \\ & T \leq t < T + 300 \\ 0 & T + 300 \leq t \end{cases}$$

ここで，T は，履歴中の $\{t, x, c\}$ の組の内の一番最後の時刻であり，$c(T)$ は 1 とした．

生産者タスクは，時刻 t に，時刻 t での合成画像を読み込む．そして画像処理の結果を $x(t)$ として *write* 操作で書き込む．消費者タスクは，時刻 t に $(t-d)$ の対象物体の位置 $x(t-d)$ を参照する．ここで，$(t-d)$ の信頼度が要求信頼度より小さい場合は *read* 操作は停止する．その結果，消費者タスクは信頼度による同期がとれている期間だけ実行される．

この 2 つのタスク間では Sieve function model に基づき情報が共有されている．ここで，要求信頼度を小さくすると情報の同期に必要となる遅延を削減することができるため，*read* 操作の停止する期間は短くなる．$d = 50\,\mathrm{ms}$，*write* 操作の周期を 250 ms にした場合の，*write* 操作と *read* 操作の関係を**図 6** に示す．

図中の矩形は，同期がとれるまで *read* 操作が停止していた期間を示す．同期がとれている場合は，*read* 操作はただちに終了するので，矩形の幅がきわめて小さくなっていることに注意してほしい．要求信頼度を上げると生産者，消費者の 2 つのタスクの間で情報の同期をとるための遅延が大きくなり，*read* 操作の待ち時間が長くなる．一方，要求信頼度を小さくすると $F_x(h,t)$ の推定結果を利用することができる．その結

果大きな遅延を生じることなく情報の同期をとることができ，*read* 操作の待ち時間は短くなる．このように，要求信頼度により情報の同期に必要となる遅延を制御することが可能である．

遅延が削減できると *read* 操作の待ち時間を短くすることができるので，スループットを向上させることが可能である．図 6 の例では，要求信頼度を 1.0 とした場合 *write* 操作と *read* 操作の比は 1 : 1 であるが，要求信頼度を 0.0 とした場合はおよそ 1 : 18 となっている．このように，要求信頼度を変化させることで，*write* 操作に対する *read* 操作の比も変化する．このように要求信頼度を下げ遅延を削減することで，スループットを向上させることが可能である．

図 7 (a) は要求信頼度，d と *read* の待ち時間の関係を示す．同期がすでにとれている過去の情報を参照することで *read* 操作の待ち時間は短くなるので d により待ち時間は変化しているが，いずれの場合にも遅延の削減の効果により要求信頼度が低いほど待ち時間は短くなっていることがわかる．

一方，図 7 (b)，(c)，(d) は，*read* 操作により得られた $x(t-d)$ の誤差を示している（(b) は最大値，(c) は最小値，(d) は平均値）．要求信頼度を下げれば誤差は大きくなり，要求信頼度を上げれば誤差は小さくなる．

これらの結果をまとめると，要求信頼度を下げれば，遅延が短くなることで *read* 操作の待ち時間が短くなるが，精度は低下する．逆に，要求信頼度を上げれば遅延が長くなることで *read* 操作の待ち時間が長くなるが，精度は向上する．すなわち，要求信頼度を調整することで，精度と遅延のトレードオフを制御できることが分かる．

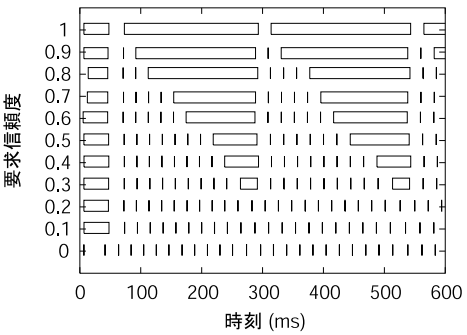


図 6 *write* 操作と *read* 操作の関係．*write* 操作は t=0, 250, 500 ms に行われている

Fig. 6 Relation between *write* operations and *read* operations. *Write* operations are executed at t=0, 250, 500 ms.

4.2 アプリケーションによる評価

次に，実際に信頼度駆動メモリを利用し，アプリケーションを作成した．これは，画像から対象人物の手領域と顔領域を抽出し対応する CG を合成するアプリケーションである．概要を**図 8** に示す．

まず過去の履歴から，あらかじめ手と顔領域それぞれの位置を予測する．この予測された位置を基準に，入力された画像から手と顔領域のそれぞれの位置を求める．顔と手の領域の抽出方法としては，一般化ハフ変換[4]によるアルゴリズムと，肌色領域のモーメント特徴量を利用するアルゴリズムの 2 つのアルゴリズムを利用した．前者は精度が良いが遅いアルゴリズムであり，後者は速いが精度の悪いアルゴリズムである．

一般化ハフ変換の処理は探索であり，より探索範囲を広げれば広げるほど探索結果の信頼度は向上する．つまり，一般化ハフ変換のタスクは Monotone model に従っている．実験で使用した計算機環境では，1 枚の画像から手領域として最もふさわしい領域を探索する場合，最大で 250 msec 必要となった．一方，肌色領域のモーメント特徴量から手や顔の領域を求める処理は 23 msec 程度で完了する．しかしながら，モーメント特徴量による方法は，手と顔の区別を厳密に行うことができず，また顔領域と手領域が重なるような場合には，対応できない．ここで，精度と遅延のトレードオフが必要となる．

2 つのタスクの使い分けには Multiple versions model を利用する．つまり，2 つの信頼度駆動メモ

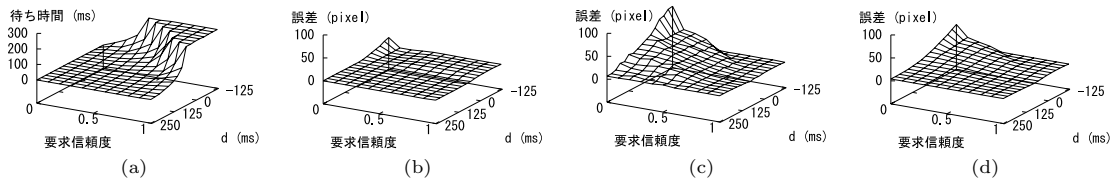


図 7 要求信頼度と待ち時間，誤差の関係．(a) 待ち時間の平均値，(b) 誤差の最小値，(c) 誤差の最大値，(d) 誤差の平均値

Fig. 7 Latency, error and required-confidence. (a) Average latency, (b) Minimum error, (c) Maximum error, and (d) Average error.

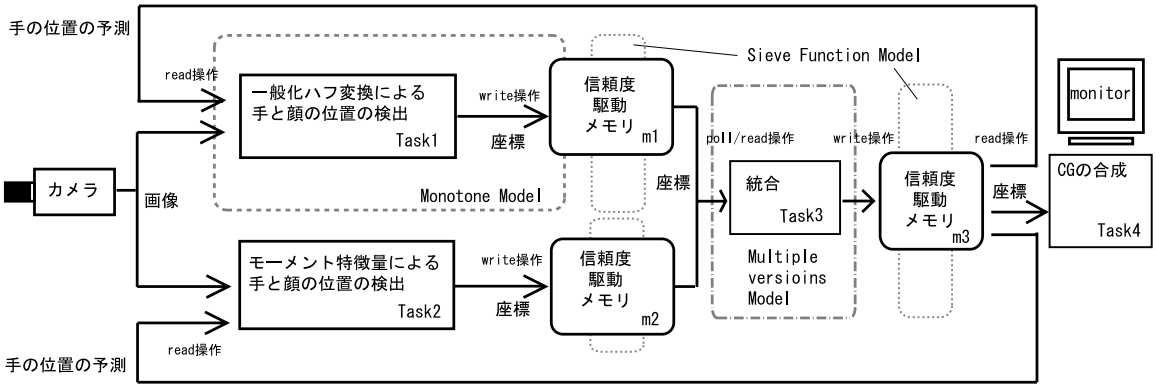


図 8 アプリケーション例
Fig. 8 Application overview.

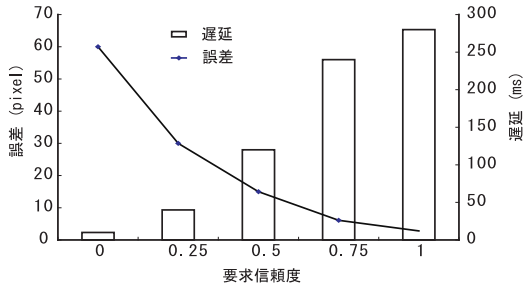


図 9 要求信頼度に対する誤差（左軸）と遅延（右軸）の関係
Fig. 9 Error, latency and required-confidence in the application.

リを用意し，*poll* 操作により信頼度が高い信頼度駆動メモリを選択し，参照する．

最後に *poll* 操作により統合された情報から，CG を合成する．*poll* 操作により得られる情報の周期と，CG を更新する周期は異なるが，ここでも単純に信頼度駆動メモリを介した通信による Sieve function model が利用可能であり，滑らかな CG を生成することが可能となった．

$d = 0$ のときの，要求信頼度とアプリケーションの出力の誤差と遅延の関係を**図 9** に示す．誤差については，アプリケーションに入力された各画像と最終的に CG として合成された出力の画像についてそれぞれ，手と顔の位置を手作業により求め，その差を誤差とした．この図が示すように，要求信頼度が高くなると誤差は小さくなるものの遅延は大きくなり，逆に要求信頼度が小さくなると誤差は大きくなるが遅延は小さくなることが分かる．このように，システムの動作特性として遅延の削減と精度の向上のどちらを優先するのかを要求信頼度により制御できることが確認できた．

5. おわりに

コンピュータビジョンの応用を考えると，様々なアルゴリズムを統合し利用する技術は必須である．共有メモリとして利用可能な信頼度駆動メモリを介した通信を行うことで，各タスクは，要求信頼度を満足する品質を持つ任意の時刻 t の情報 $x(t)$ を共有することが可能となった．さらに，信頼度駆動メモリは，信頼度駆動の概念による同期機構を提供する．この同期機構により，Imprecise 計算モデルに従ったシステム構成が簡潔に実現可能となり，精度と遅延のトレードオフの問題を，単純な信頼度の同期の問題として解決できるようになった．

このように，信頼度駆動メモリは実時間ビジョンシステムにとって簡潔かつ有用なタスク間通信機構である．本論文では，基本的なアイデアとその有効性をシミュレーションと簡単なアプリケーション例による実験結果で示した．

今後の課題としては，信頼度に基づいたスケジューラの作成，分散環境への本格的な対応があげられる．本論文でも述べたように，我々は各タスクの周期性を考えることでさらに遅延時間の削減が可能であると考えている．そのためには，タスクや通信の優先順位を考慮したスケジューラが必要である．また，信頼度駆動メモリを，分散共有メモリに適用することも可能である．現在これらの作業と，具体的なビジョンアプリケーションを用いた評価を進めている．

謝辞 本研究の一部は，通信・放送機構「次世代インテリジェント・マルチメディア情報通信網の基盤技術に関する研究（No.10080）」および，科学研究費補助金基盤研究課題番号 14380167 の補助を受けた．

参 考 文 献

1) Liu, J.W.S., Shih, W., Lin, K., Bettati, R. and Chung, J.: Imprecise Computations, *Proc. 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) Ballard, D.H.: Generalizing the Hough transformation to detect arbitrary shapes, *PR*, Vol.13, No.2, pp.111–122 (1981).

(平成 14 年 9 月 19 日受付)
(平成 15 年 9 月 5 日採録)

吉本　廣雅

平成 8 年九州大学工学部電気情報工学科卒業．平成 14 年九州大学大学院システム情報科学府知能システム学専攻修士課程修了．現在，同博士後期課程在学中．

有田　大作（正会員）

平成 4 年京都大学工学部情報工学科卒業．平成 10 年九州大学大学院システム情報科学研究科博士後期課程修了．同年，同（現大学院システム情報科学研究院）助手．博士（工学）．文書画像処理，画像処理における知識獲得，実時間並列画像処理，会話情報学の研究に従事．電子情報通信学会，映像情報メディア学会各会員．

谷口倫一郎（正会員）

昭和 53 年九州大学工学部情報工学科卒業．昭和 55 年九州大学大学院工学研究科修士課程修了．同年九州大学大学院総合理工学研究科助手．平成元年同助教授．平成 8 年九州大学大学院システム情報科学研究科（現大学院システム情報科学研究院）教授．工学博士．画像処理，コンピュータビジョン，ヒューマンインタフェース，並列処理等に関する研究に従事．本会論文賞（1993），同坂井記念特別賞（1995）を受賞．