

(19) 日本国特許庁(JP)

(12) 公開特許公報(A)

(11) 特許出願公開番号

特開2017-191405

(P2017-191405A)

(43) 公開日 平成29年10月19日(2017. 10. 19)

(51) Int.Cl.	F I	テーマコード (参考)
G06F 17/10 (2006.01)	G06F 17/10 Z	5B013
G06T 1/20 (2006.01)	G06T 1/20 B	5B056
G06F 9/38 (2006.01)	G06F 9/38 370C	5B057

審査請求 未請求 請求項の数 4 O L (全 12 頁)

(21) 出願番号	特願2016-79586 (P2016-79586)	(71) 出願人	000004260
(22) 出願日	平成28年4月12日 (2016. 4. 12)		株式会社デンソー
		(74) 代理人	110000567
			特許業務法人 サトー国際特許事務所
		(72) 発明者	野村 尚利
			愛知県刈谷市昭和町1丁目1番地 株式会
			社デンソー内
		Fターム(参考)	5B013 DD03
			5B056 AA06 BB00
			5B057 CA08 CA13 CB08 CB13 CH04
			CH08

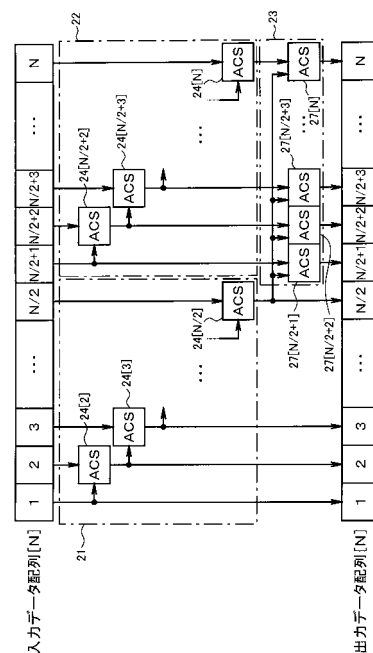
(54) 【発明の名称】 演算装置

(57) 【要約】

【課題】 演算の精度を良好に維持しつつ、演算の高速化を実現する。

【解決手段】 演算装置であるコスト補正処理回路は、複数の入力データ [1] ~ [N] のそれぞれに対し、前段の演算結果を用いる加算比較選択演算を入力データ配列の順序に従い実行する。コスト補正処理回路は、入力データ [1] ~ [N] に対する一連の演算処理を上記順序に沿って前半と後半の2つに分割し、それぞれを並列に実行する分割部21、22と、後半の分割部22の各段から出力される演算結果のそれぞれに対し前半の分割部21の最終段から出力される演算結果を一括比較して統合する統合部23と、を備える。

【選択図】 図4



【特許請求の範囲】

【請求項 1】

複数の入力データのそれぞれに対し、前段の演算結果を用いる演算処理を所定の順序に従い実行する演算装置（１２）であって、

前記演算処理は、前記複数の入力データのうちいずれか１つの入力データと、前段の演算結果に所定の定数を加算したデータとを比較し、それら各データのうち予め定めた条件を満たす値を選択して演算結果として出力する加算比較選択演算であり、

前記複数の入力データに対する一連の前記演算処理を、前記順序に沿って複数の分割し、それぞれを並列に実行する分割部（２１、２２、３１～３４）と、

前記順序が隣り合う２つの分割部のうち前記順序が後の分割部の各段から出力される演算結果のそれぞれに対し前記順序が前の分割部の最終段から出力される演算結果を一括比較して統合する統合処理を実行する統合部（２３、３５、４１）と、

を備える演算装置。

【請求項 2】

前記分割部（２１、２２）は、前記複数の入力データに対する一連の前記演算処理を、前記順序の前半および後半の２つに分割し、それぞれを並列に実行するものであり、

前記統合部（２３）は、後半の前記分割部の各段から出力される演算結果のそれぞれに対し、前半の前記分割部の最終段から出力される演算結果を一括比較して統合する請求項 1 に記載の演算装置。

【請求項 3】

前記分割部（３１～３４）は、前記複数の入力データに対する一連の前記演算処理を、前記順序に沿って３つ以上に分割し、それぞれを並列に実行するものであり、

前記統合部（３５）は、前記順序が３番目以降の分割部の各段から出力される演算結果のそれぞれに対し、当該分割部より前記順序が１つ前の分割部から出力されて前記統合が行われた後の演算結果を一括比較して統合する請求項 1 に記載の演算装置。

【請求項 4】

前記分割部（３１～３４）は、前記複数の入力データに対する一連の前記演算処理を、前記順序に沿って３つ以上に分割し、それぞれを並列に実行するものであり、

前記統合部（４１）は、前記順序が隣り合う２つの分割部を組に対して前記統合処理を実行し、前記統合処理が実行された組のうち順序が隣り合う２つの組に対してさらに前記統合処理を実行する請求項 1 に記載の演算装置。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、加算比較選択演算を繰り返し実行する演算装置に関する。

【背景技術】

【０００２】

近年、自動車の事故防止や自動運転化のために、車載カメラによる測距や認識処理のニーズが高まっている。車載向けの画像処理装置は組み込みシステムのため、限られた性能およびリソースで要求時間内での処理を実現する必要がある。例えば、ステレオカメラは原理的に模様の少ない物体の測距が難しい。そのため、グローバル法という視差画像全体の滑らかさを考慮する手法を導入することにより、弱パターンである路面などもロバストに検出することができる手法が近年開発されている。このような方法の主なものとして、Vi terbi（以下、ビタビと呼ぶ）アルゴリズムと呼ばれる動的計画法の一種により、全体のコストが最小になる視差の組み合わせを決定するものがある。

【０００３】

ビタビアルゴリズムの計算量としては、その遷移状態、つまりノードの数が増えると、指数関数的に処理量が増える。そして、ステレオカメラの場合、遷移状態数が視差の数（通常、数十～数百以上の数）だけあるため、計算量が大きくなる。このような課題に対して、アルゴリズムに着目し、既に計算した隣の遷移状態の結果を再利用することにより、

従来ではノード数の2乗に比例していた計算量を、ノード数の2倍程度の計算量に抑えることを実現する手法が考えられている（例えば、特許文献1参照）。このような手法によれば、処理時間の短縮が可能になるが、ステレオカメラの数百万画素にわたって上記処理を車載のような組み込み環境で実現するためには、更なる高速化が必要となる。

【先行技術文献】

【特許文献】

【0004】

【特許文献1】特開2015-114269号公報

【特許文献2】特開2014-045480号公報

【発明の概要】

10

【発明が解決しようとする課題】

【0005】

ビタビ演算の高速化の課題に対し、ビタビ演算の基本処理要素である加算・比較・選択（以下、ACSと呼ぶ）回路を直列に接続した専用回路が提案されている（例えば、特許文献2参照）。この手法によれば、単一のACS回路を再帰的に用いる場合に比べ、クロックの受け渡しでの処理時間のオーバーヘッドを減らせる分だけ高速化が期待できる。しかし、この場合、通過するACS回路の段数は少なくなっておらず、スループットの向上には限界がある。

【0006】

本発明は上記事情に鑑みてなされたものであり、その目的は、演算の精度を良好に維持しつつ、演算の高速化を実現することができる演算装置を提供することにある。

20

【課題を解決するための手段】

【0007】

請求項1に記載の演算装置（12）は、複数の入力データのそれぞれに対し、前段の演算結果を用いる演算処理を所定の順序に従い実行する。その演算処理は、複数の入力データのうちいずれか1つの入力データと、前段の演算結果に所定の定数を加算したデータとを比較し、それら各データのうち予め定めた条件を満たす値を選択して演算結果として出力する加算比較選択演算である。上記演算装置は、複数の入力データに対する一連の演算処理を上記順序に沿って複数に分割し、それぞれを並列に実行する分割部（21、22、31～34）と、上記順序が隣り合う2つの分割部のうち順序が後の分割部の各段から出力される演算結果のそれぞれに対し順序が前の分割部の最終段から出力される演算結果を一括比較して統合する統合部（23、35、41）と、を備える。

30

【0008】

このような構成によれば、一連の演算処理を複数に分割して並列に実行するので、一連の演算処理を逐次的に処理する場合に比べ、演算処理に要する時間が短縮される。また、統合部を設けたことにより、順序が後の分割部から出力される演算結果についても、順序が前の分割部における演算結果を考慮したものとなり、演算が不正確になることはない。したがって、本手段によれば、演算の精度を良好に維持しつつ、演算の高速化を実現することができる。

【図面の簡単な説明】

40

【0009】

【図1】第1実施形態に係る距離検出装置の構成を模式的に示す図

【図2】処理部の具体的な構成を模式的に示す図

【図3】距離演算処理の内容を模式的に示すフローチャート

【図4】コスト補正処理回路の構成を模式的に示す図

【図5】分割部におけるACS回路の構成を模式的に示す図

【図6】統合部におけるACS回路の構成を模式的に示す図

【図7】第2実施形態に係るコスト補正処理回路の第1構成例を模式的に示す図

【図8】第2実施形態に係るコスト補正処理回路の第2構成例を模式的に示す図

【発明を実施するための形態】

50

【0010】

以下、本発明の複数の実施形態について図面を参照して説明する。なお、各実施形態において実質的に同一の構成には同一の符号を付して説明を省略する。

(第1実施形態)

以下、本発明の第1実施形態について図1～図6を参照して説明する。

図1に示す距離検出装置1は、複数の撮像画像の視差を検出することによって撮像画像中の各点までの距離を検出する装置である。距離検出装置1では、動的計画法であるビタビアルゴリズムを複数の方向に適用することで複数の撮像画像を構成する画素間の対応関係、つまり視差を高精度に求めることができるように配慮されている。

【0011】

10

距離検出装置1は、乗用車などの車両に搭載されており、2つの撮像部2、3、処理部4および車両制御部5を備えている。なお、撮像部2、3は、2つに限らずともよく、例えば3つ以上設けられていてもよい。撮像部2、3は、それぞれ車両の進行方向が撮像範囲内となる周知のカメラとして構成されている。撮像部2、3は、ステレオカメラを構成しており、周期的に同時に撮像を行うように設定されている。

【0012】

処理部4は、撮像部2、3により撮像された撮像画像を取得し、後述する各種処理を実行する。車両制御部5は、処理部4による処理結果を用いて車両を制御する処理を実行する。例えば、車両制御部5は、撮像部2、3による撮像範囲内の各点における距離の情報を処理部4から取得し、その距離の情報に基づいて物標位置および相対速度を認識する。そして、車両制御部5は、物標が走行に支障をきたすおそれがある場合、走行軌道を変更するなどの車両制御を実行する。

20

【0013】

処理部4の機能は、例えば、図2に示すような構成により実現することができる。すなわち、この場合、処理部4は、バス6を介して通信可能に接続されたCPU7、DMAコントローラ8、メインメモリ9および専用アクセラレータ10により構成される。専用アクセラレータ10は、プログラム制御部11、コスト補正処理回路12、その他演算回路13、ロードストア制御回路14、ローカルメモリ15およびデータレジスタ16を備えている。

【0014】

30

上記構成の基本的な動作および機能は、次の通りである。すなわち、ホストCPU7は、専用アクセラレータ10に対し、対象とする処理依頼を通知する。また、メインメモリ9から処理対象のデータがDMAコントローラ8を介して専用アクセラレータ10のローカルメモリ15に順次転送される。

【0015】

プログラム制御部11は、CPU7からの処理依頼に基づいて対象の処理プログラムを読み出し、各回路に対して対象処理や対象レジスタなどを指定する制御信号を送信する。ロードストア制御回路14は、上記制御信号に基づいてローカルメモリ15から複数データを単位とした配列をデータレジスタにロードまたはストアする。

【0016】

40

コスト補正処理回路12およびその他演算回路13は、上記制御信号に基づいて対象レジスタのデータを読み出して処理を行い、レジスタに書き戻す。プログラム制御部11は、処理プログラムが完了すると、CPU7およびDMAコントローラ8に完了通知を転送する。また、ローカルメモリ15から完了した処理データがDMAコントローラ8を介してメインメモリ9に転送される。その後、ホストCPU7は、次の処理を判断し、次の処理依頼を専用アクセラレータ10に転送する。処理部4は、以上のような動作を繰り返すことにより、各種の処理を実行する。

【0017】

本実施形態の距離検出装置1において、処理部4は、特開2015-114269号公報（以下、文献1とも呼ぶ）に開示された距離演算処理と同様の処理を行う。すなわち、

50

図 3 に示すように、まず、ステップ S 1 では、撮像部 2、3 により撮像された撮像画像が取得される。続くステップ S 2 では、撮像画像のレンズによる画像の歪や姿勢のずれを補正するなどの処理が行われ、撮像された画像が平行化される。

【0018】

ステップ S 3 では、対応点探索処理が実施される。対応点探索処理は、例えば撮像部 2 による撮像画像である基準画像を構成する各画素と、例えば撮像部 3 による撮像画像である比較画像を構成する各画素との対応関係、つまり同じものが写っている場所に関連付ける処理である。対応点探索処理の具体的な内容は、文献 1 に開示されているため、ここでの説明は省略する。続いて、ステップ S 4 では、各画素における対応点の視差に応じて各画素に写っている物標までの距離が算出される。そして、ステップ S 5 では、これらの画素と距離との対応データが車両制御部 5 に対して出力される。

10

【0019】

ただし、本実施形態では、対応点探索処理に対し、以下に説明するような工夫を加えることで、演算の更なる高速化を実現している。すなわち、文献 1 の対応点探索処理では、1 つ前までの最小コスト結果を再利用することで総当たり計算を二項比較の反復に直し、処理負荷の削減が図られている。そして、本願の発明者は、このような文献 1 では「 $s(u, v) = u - v$ 」としているが、「 $s(u, v) = \lambda |u - v|$ 」であれば線形性が保たれることを見出した。ただし、 λ は所定の定数である。この場合、前の結果に λ を加算して次のノードの値と比較し、小さいほうの値を次の結果とする、といったことが繰り返される。つまり、この場合、ACS 演算が基本の繰り返し単位となる。

20

【0020】

ここで、ACS 演算を行う ACS 回路を直列に接続する従来技術を用いると、1 つ前の結果を用いる演算および選択が直列に反復処理される、つまり逐次的に処理されるため、多段接続による回路遅延などに起因する速度低下の影響が大きい。そこで、このような逐次処理を分割して並列に実行することが考えられる。しかし、演算結果がその後の演算に依存するような逐次処理を、単純に分割して並列化すると、計算が不正確になり、精度が低下するといった懸念がある。

【0021】

これらの点を考慮し、本実施形態では、遷移コストの線形性を導入することを前提として、逐次処理を行う回路を複数のブロックに分割して並列動作させ、それら並列動作させた各ブロックにおける結果を一括比較して統合するようにした。具体的には、本実施形態のコスト補正処理回路 12 は、図 4 に示すような構成を有している。なお、コスト補正処理回路 12 は、複数の入力データのそれぞれに対し、前段の演算結果を用いる演算処理を所定の順序に従い実行する演算装置に相当する。

30

【0022】

コスト補正処理回路 12 は、逐次処理を行う回路を前半の分割部 21 と後半の分割部 22 とに分割して並列処理し、最後に統合部 23 により各段の結果を統合する。なお、以下では、コスト補正処理回路 12 に入力される入力データ配列の各データのことを入力データ [1]，入力データ [2]，…，入力データ [N] と呼び、コスト補正処理回路 12 から出力される出力データ配列の各データのことを出力データ [1]，出力データ [2]，…，出力データ [N] と呼ぶこととする。また、本実施形態では、これらのデータ数 N は、偶数となっている。

40

【0023】

前半の分割部 21 は、入力データ [2] ～ [N/2] をそれぞれ入力データとする ACS 回路 24 [2] ～ 24 [N/2] を備えている。また、後半の分割部 22 は、入力データ [N/2 + 2] ～ [N] をそれぞれ入力データとする ACS 回路 24 [N/2 + 2] ～ 24 [N] を備えている。なお、以下では、ACS 回路 24 [2] ～ ACS 回路 24 [N] のことを ACS 回路 24 と総称することがある。

【0024】

ACS 回路 24 は、図 5 に示すような構成となっている。すなわち、ACS 回路 24 は

50

、加算器 25 および比較選択器 26 を備えている。加算器 25 は、前段から出力される演算結果に定数 λ を加算する。なお、本実施形態では、定数 λ は正の値となっており、加算器 25 は純粋に加算を行うものである。また、定数 λ の値は、レジスタ設定などにより適宜変更することが可能となっている。比較選択器 26 は、加算器 25 の出力データと本段の入力データとを比較し、それら各データのうち予め定めた条件を満たす値を選択して本段の演算結果として出力する。具体的には、比較選択器 26 は、各データのうちいずれか小さいほうの値を選択して本段の演算結果として出力する。

【0025】

例えば、ACS 回路 24 [2] は、入力データ [1] に定数 λ を加算して入力データ [2] と比較し、小さいほうの値を本段の演算結果として出力するとともに、次段の ACS 回路 24 [3] に対し前段の演算結果として出力する。そして、次段の ACS 回路 24 [3] は、ACS 回路 24 [2] から出力された前段の演算結果に定数 λ を加算して入力データ [3] と比較し、いずれか小さいほうの値を本段の演算結果として出力するとともに、次段の ACS 回路 24 [4] に対し前段の演算結果として出力する。

【0026】

また、ACS 回路 24 [$N/2 + 2$] は、入力データ [$N/2 + 1$] に定数 λ を加算して入力データ [$N/2 + 2$] と比較し、小さいほうの値を本段の演算結果として統合部 23 に出力するとともに、次段の ACS 回路 24 [$N/2 + 3$] に対し前段の演算結果として出力する。そして、次段の ACS 回路 24 [$N/2 + 3$] は、ACS 回路 24 [$N/2 + 2$] から出力された前段の演算結果に定数 λ を加算して入力データ [$N/2 + 3$] と比較し、いずれか小さいほうの値を本段の演算結果として統合部 23 に出力するとともに、次段の ACS 回路 24 [$N/2 + 4$] に対し前段の演算結果として出力する。

【0027】

このような構成において、入力データ [1] がそのまま出力データ [1] となり、前半の分割部 21 が備える ACS 回路 24 [2] ~ 24 [$N/2$] から出力される演算結果が、それぞれ出力データ [2] ~ [$N/2$] となる。ただし、ACS 回路 24 [$N/2$] から出力される演算結果は、統合部 23 にも与えられる。

【0028】

統合部 23 は、分割部 22 の各段の出力をそれぞれ入力データとする ACS 回路 27 [$N/2 + 1$] ~ 27 [N] を備えている。なお、以下では、ACS 回路 27 [$N/2 + 1$] ~ ACS 回路 27 [N] のことを ACS 回路 27 と総称することがある。

【0029】

ACS 回路 27 は、図 6 に示すような構成となっている。すなわち、ACS 回路 27 は、加算器 28 および比較選択器 29 を備えている。加算器 28 は、前半の最終段の結果、つまり ACS 回路 24 [$N/2$] から出力される演算結果に α を加算する。 α は、後半における順番を表す数に定数 λ を乗じた値である。

【0030】

例えば、ACS 回路 27 [$N/2 + 1$] の場合、後半における順番が「1」であるため「 $\alpha = 1 \times \lambda$ 」となり、ACS 回路 27 [$N/2 + 2$] の場合、後半における順番が「2」であるため「 $\alpha = 2 \times \lambda$ 」となり、ACS 回路 27 [N] の場合、後半における順番が「 $N/2$ 」であるため「 $\alpha = (N/2) \times \lambda$ 」となる。したがって、ACS 回路 27 [X] とすると、 α は下記 (1) で表される。

$$\alpha = (X - N/2) \times \lambda \quad \cdots (1)$$

【0031】

比較選択器 29 は、加算器 28 の出力データと分割部 22 の各段から出力される演算結果とを比較し、いずれか小さいほうの値を後半の各段の最終結果として出力する。例えば、ACS 回路 27 [$N/2 + 1$] は、前半の最終段の演算結果に α ($= 1 \times \lambda$) を加算して入力データ [$N/2 + 1$] と比較し、小さいほうの値を本段の最終結果、つまり出力データ [$N/2 + 1$] として出力する。また、ACS 回路 27 [$N/2 + 2$] は、前半の最終段の演算結果に α ($= 2 \times \lambda$) を加算して ACS 回路 24 [$N/2 + 2$] から出力され

10

20

30

40

50

る演算結果と比較し、小さいほうの値を本段の最終結果、つまり出力データ $[N/2 + 2]$ として出力する。

【0032】

このようなACS回路27を有する統合部23は、分割部21の最終段の演算結果と比例関係にあるデータ（＝加算器28の出力データ）と、分割部22の各段から出力される演算結果とを比較し、いずれか小さいほうの値を一連のACS演算処理における後半の各段の最終結果として出力するようになっている。つまり、統合部23は、後半の分割部22の各段から出力される演算結果のそれぞれに対し前半の分割部21の最終段から出力される演算結果を一括比較して統合するようになっている。

【0033】

以上説明した本実施形態によれば、次のような効果が得られる。

本実施形態のコスト補正処理回路12は、入力データ配列の各入力データ $[1] \sim [N]$ のそれぞれに対し、前段の演算結果を用いるACS演算を入力データ配列の順序に従い実行する。そして、コスト補正処理回路12は、複数の入力データ $[1] \sim [N]$ に対する一連のACS演算処理を上記順序に沿って前半と後半に分割し、それぞれを並列に実行する分割部21、22と、後半の分割部22の各段から出力される演算結果のそれぞれに対し前半の分割部21の最終段から出力される演算結果を一括比較して統合する統合部23と、を備えている。

【0034】

このような構成によれば、一連のACS演算処理を2つに分割して並列に実行するので、一連のACS演算処理を逐次的に処理する場合に比べ、回路遅延の短縮などを期待することができ、その結果、演算処理に要する時間が約半分程度に短縮される。しかも一連のACS演算処理を単純に並列化しただけでなく、統合部23を設けているため、後半の分割部22から出力される演算結果についても、前半の分割部21での演算結果を考慮したものとなり、演算が不正確になるおそれはない。したがって、本実施形態によれば、演算の精度を良好に維持しつつ、演算の高速化を実現することができる。このような演算の高速化、つまり処理時間の削減による効果は、車載のような組み込み環境において、一層有益なものとなる。

【0035】

（第2実施形態）

第1実施形態では、コスト補正処理回路12は、複数の入力データ $[1] \sim [N]$ に対する一連のACS演算処理を前半と後半の2つに分割していたが、この分割数は3つ以上でもよい。本実施形態では、一連のACS演算処理を4つに分割した2つの構成例について図7および図8を参照して説明する。

【0036】

<第1構成例>

図7に示すように、第1構成例では、一連のACS演算処理を入力データ配列の順序に沿って4つの分割部31～34に分割して並列処理し、最後に統合部35により各段の結果を統合する。分割部31は、入力データ $[2] \sim [N/4]$ をそれぞれ入力データとするACS回路24 $[2] \sim 24 [N/4]$ を備えている。分割部32は、入力データ $[N/4 + 2] \sim [N/2]$ をそれぞれ入力データとするACS回路24 $[N/4 + 2] \sim 24 [N/2]$ を備えている。

【0037】

分割部33は、入力データ $[N/2 + 2] \sim [3N/4]$ をそれぞれ入力データとするACS回路24 $[N/2 + 2] \sim 24 [3N/4]$ を備えている。分割部34は、入力データ $[3N/4 + 2] \sim [N]$ をそれぞれ入力データとするACS回路24 $[3N/4 + 2] \sim 24 [N]$ を備えている。

【0038】

このような構成において、入力データ $[1]$ がそのまま出力データ $[1]$ となり、分割部31が備えるACS回路24 $[2] \sim 24 [N/4]$ から出力される演算結果が、それ

10

20

30

40

50

ぞれ出力データ〔2〕～〔 $N/4$ 〕となる。ただし、ACS回路24〔 $N/4$ 〕から出力される演算結果は、統合部35にも与えられる。

【0039】

統合部35は、第1統合ブロック35a、第2統合ブロック35bおよび第3統合ブロック35cを備えている。第1統合ブロック35aは、分割部32の各段の出力をそれぞれ入力データとするACS回路27〔 $N/4+1$ 〕～27〔 $N/2$ 〕を備えている。ACS回路27〔 $N/4+1$ 〕～27〔 $N/2$ 〕の加算器28は、分割部31の最終段の結果、つまりACS回路24〔 $N/4$ 〕から出力される演算結果に α を加算するようになっている。したがって、第1統合ブロック35aは、分割部31の最終段の演算結果と比例関係にあるデータと、分割部32の各段から出力される演算結果とを比較し、いずれか小さいほうの値を選択して出力データ〔 $N/4+1$ 〕～〔 $N/2$ 〕として出力する。

10

【0040】

第2統合ブロック35bは、分割部33の各段の出力をそれぞれ入力データとするACS回路27〔 $N/2+1$ 〕～27〔 $3N/4$ 〕を備えている。ACS回路27〔 $N/2+1$ 〕～27〔 $3N/4$ 〕の加算器28は、第1統合ブロック35aの最終段の結果、つまりACS回路27〔 $N/2$ 〕から出力される演算結果に α を加算するようになっている。したがって、第2統合ブロック35bは、分割部32の最終段から出力されて第1統合ブロック35aを介して統合が行われた後の演算結果である出力データ〔 $N/2$ 〕と比例関係にあるデータと、分割部33の各段から出力される演算結果とを比較し、いずれか小さいほうの値を選択して出力データ〔 $N/2+1$ 〕～〔 $3N/4$ 〕として出力する。

20

【0041】

第3統合ブロック35cは、分割部34の各段の出力をそれぞれ入力データとするACS回路27〔 $3N/4+1$ 〕～27〔 N 〕を備えている。ACS回路27〔 $3N/4+1$ 〕～27〔 N 〕の加算器28は、第2統合ブロック35bの最終段の結果、つまりACS回路27〔 $3N/4$ 〕から出力される演算結果に α を加算するようになっている。したがって、第3統合ブロック35cは、分割部33の最終段から出力されて第2統合ブロック35bを介して統合が行われた後の演算結果である出力データ〔 $3N/4$ 〕と比例関係にあるデータと、分割部34の各段から出力される演算結果とを比較し、いずれか小さいほうの値を選択して出力データ〔 $3N/4+1$ 〕～〔 N 〕として出力する。

【0042】

30

このように、第1構成例の統合部35は、順序が1番目、2番目の分割部31、32を組にして統合処理を行い、順序が3番目以降の分割部33、34の各段から出力される演算結果のそれぞれに対し、それらより順序が1つ前の分割部から出力されて統合処理が行われた後の演算結果を一括比較して統合し、最終結果として出力するようになっている。なお、この場合、統合部35では、「(1)第1統合ブロック35aによる統合→(2)第2統合ブロック35bによる統合→(3)第3統合ブロック35cによる統合」という順に、3段階で統合処理が行われることになる。

【0043】

<第2構成例>

図8に示すように、第2構成例では、第1構成例と同様に、一連のACS演算処理を入力データ配列の順序に沿って4つの分割部31～34に分割して並列処理し、最後に統合部41により各段の結果を統合する。統合部41は、第1統合ブロック41a、第2統合ブロック41bおよび第3統合ブロック41cを備えている。

40

【0044】

第1統合ブロック41aは、第1構成例の第1統合ブロック35aと同様の構成となっている。したがって、第1統合ブロック41aは、分割部31の最終段の演算結果と比例関係にあるデータと、分割部32の各段から出力される演算結果とを比較し、いずれか小さいほうの値を選択して出力データ〔 $N/4+1$ 〕～〔 $N/2$ 〕として出力する。

【0045】

第2統合ブロック41bは、分割部34の各段の出力をそれぞれ入力データとするACS

50

S回路27[$3N/4+1$]~27[N]を備えている。ACS回路27[$3N/4+1$]~27[N]の加算器28は、分割部33の最終段の結果、つまりACS回路24[$3N/4$]から出力される演算結果に α を加算するようになっている。したがって、第2統合ブロック41bは、分割部33の最終段の演算結果と比例関係にあるデータと、分割部34の各段から出力される演算結果とを比較し、いずれか小さいほうの値を選択して出力する。

【0046】

第3統合ブロック41cは、分割部33の各段の出力をそれぞれ入力データとするACS回路27[$N/2+1$]~27[$3N/4$]と、第2統合ブロック41bの各段の出力をそれぞれ入力データとするACS回路42[$3N/4+1$]~42[N]と、を備えている。なお、ACS回路42[$3N/4+1$]~42[N]は、ACS回路27と同様の構成である。また、以下では、ACS回路42[$3N/4+1$]~42[N]のことをACS回路42と総称することがある。

【0047】

ACS回路27[$N/2+1$]~27[$3N/4$]およびACS回路42[$3N/4+1$]~42[N]の加算器28は、第1統合ブロック41aの最終段の結果、つまりACS回路27[$N/2$]から出力される演算結果に α を加算するようになっている。したがって、第3統合ブロック41cは、分割部32の最終段から出力されて第1統合ブロック41aを介して統合が行われた後の演算結果である出力データ[$N/2$]と比例関係にあるデータと、分割部33の各段から出力される演算結果とを比較し、いずれか小さいほうの値を選択して出力データ[$N/2+1$]~[$3N/4$]として出力する。また、第3統合ブロック41cは、出力データ[$N/2$]と比例関係にあるデータと、分割部34の各段から出力されて第2統合ブロック41bを介して統合が行われた後の演算結果とを比較し、いずれか小さいほうの値を選択して出力データ[$3N/4+1$]~[N]として出力する。

【0048】

このように、第2構成例の統合部41は、順序が隣り合う2つの分割部31、32を組にして統合処理を行うとともに、順序が隣り合う2つの分割部33、34を組にして統合処理を行う。そして、統合部41は、統合処理が行われた2つの組に対してさらに統合処理を行い、最終結果として出力するようになっている。なお、この場合、統合部41では、「(1)第1統合ブロック41aによる統合および第2統合ブロック41bによる統合→(2)第3統合ブロック41cによる統合」という順に、2段階で統合処理が行われることになる。

【0049】

以上説明した本実施形態によっても、第1実施形態と同様に、演算の精度を良好に維持しつつ、演算の高速化を実現することができる。なお、第1構成例および第2構成例は、それぞれにメリットがある。すなわち、第1構成例の統合部35は、第2構成例の統合部41に比べ、ACS回路の数を少なくできるため、その回路規模を小さくすることができる。また、第2構成例の統合部41は、第1構成例の統合部35に比べ、ACS回路を通過する段数を小さく抑えることができるため、演算処理に要する時間を更に短縮することができる。したがって、一連のACS演算処理の分割数を4以上にする場合、このような点を考慮したうえで、回路規模や処理時間などの仕様を満たすように、最適な構成を用いればよい。

【0050】

(その他の実施形態)

なお、本発明は上記し且つ図面に記載した各実施形態に限定されるものではなく、その要旨を逸脱しない範囲で任意に変形、組み合わせ、あるいは拡張することができる。

上記各実施形態では、ACS演算は「加算→最小比較」であったが、「減算→最大比較」としてもよい。すなわち、上記各実施形態では定数 λ を正の値としていたが、これを負の値としてもよい。そうすると、加算器25、28は、実質的に減算を行うものとなる。

この場合、比較選択器 26、29 は、2つの入力値のうち、いずれか大きいほうの値を選択して演算結果として出力すればよい。つまり、この場合、いずれか大きい値が「予め定めた条件を満たす値」となる。

上記各実施形態では、一連の A C S 演算処理を分割する際、それぞれの分割部を等分するようにしていたが、必ずしも等分する必要はない。また、上記各実施形態では、入力データ配列のデータ数 N は偶数としていたが、データ数 N は奇数でもよい。

【0051】

上記各実施形態では一連の A C S 演算処理を、入力データ配列の順序に沿って複数に分割していたが、これに限らずともよく、所定の順序に沿って複数に分割すればよい。

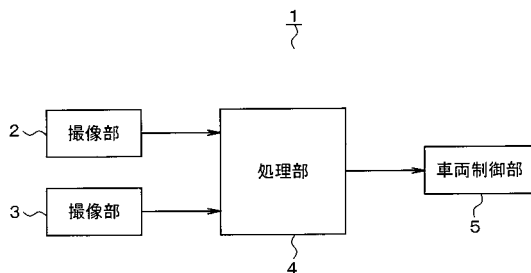
統合部 23、35、41 の A C S 回路のうち、後段部分を省略することも可能である。例えば、統合部における所定の A C S 回路での比較選択において加算器側のデータが最小値ではないとされて一旦棄却された場合、その後段の A C S 回路での比較選択においても必ず加算器側のデータは棄却される。そこで、予め実験やシミュレーションなどで、統合部の後半のある所で非常に高い確率で加算器側のデータが棄却されることが判明していれば、それ以降は A C S 回路を省略し、前段の演算結果を一括比較して統合する処理を行わないことも可能である。このようにすれば、回路規模を小さくできるという効果が得られる。なお、このようにした場合でも、大きく演算の精度が低下することはない。

【符号の説明】

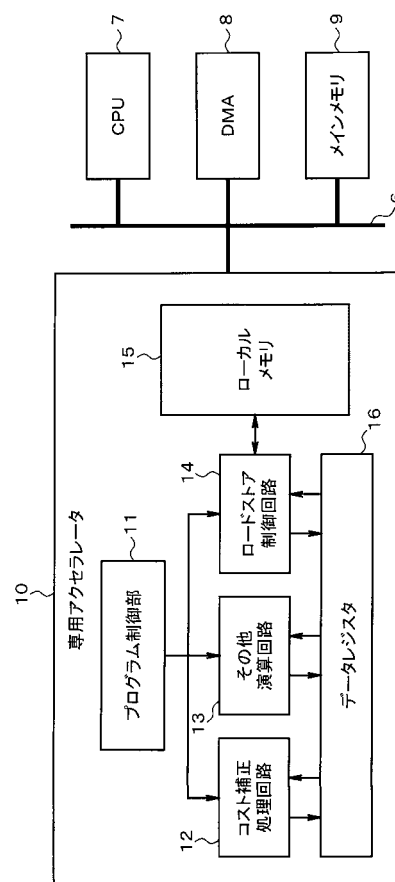
【0052】

12…コスト補正処理回路、21、22、31～34…分割部、23、35、41…統合部。

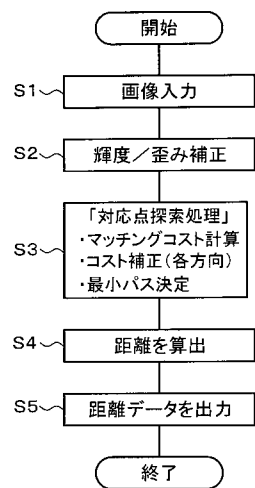
【図 1】



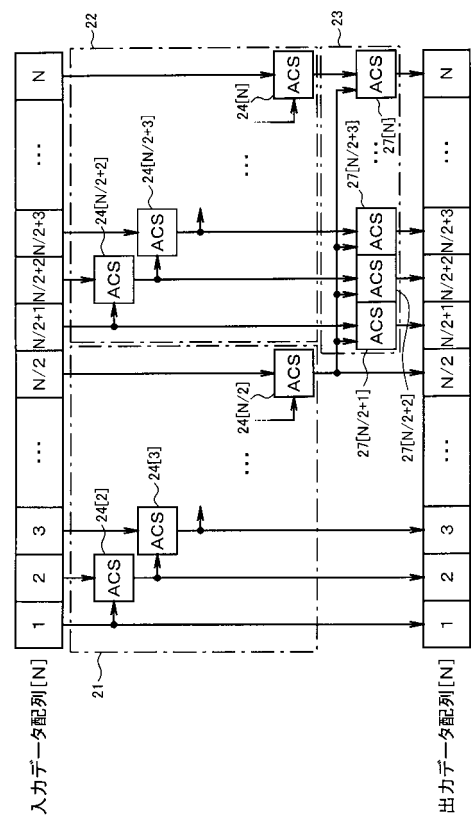
【図 2】



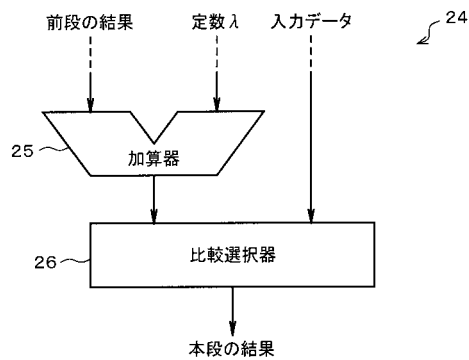
【 図 3 】



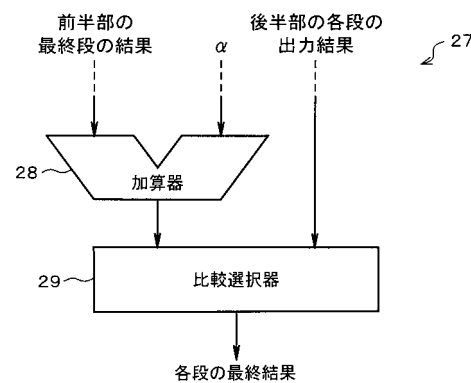
【 図 4 】



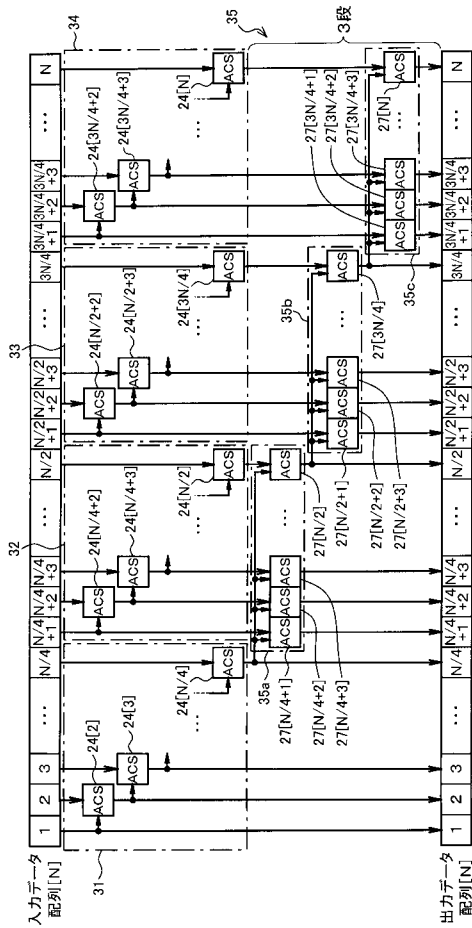
【 図 5 】



【 図 6 】



【図 7】



【図 8】

