モフォロジー処理のための進化計算による自動最適化手法とFPGAを用いた高速ラベリングの応用可能性について

星野　孝総

（受領日：2010年5月6日）

高知工科大学システム工学群
〒782-8502 高知県香美市土佐山田町宮ノ口185

E-mail:hoshino.yukinobu@kochi-tech.ac.jp

要約：本稿では画像処理システムを用いた基盤検査を念頭に置き、そのシステムのハードウェア化を容易にしたFPGAをとりあげ、実装実験を通じてその応用可能性について検証する。現在、FPGAを用いた高速なリアルタイム処理はすでに多くに実用例が報告され、生産技術を支える基盤技術として定着しつつある。また、多くの企業の生産現場でも用いられており、エンジニアの育成が工学系大学に化せられた課題とされている。これは画像処理に関する専門知識とコンピュータプログラムとハードウェア設計の両面から設計しなくてはならず。これらを支援するシステム研究に我々は着目した。そこで、進化計算を用いオフラインで最適な制御パラメータを計算するアプローチを述べ、さらにFPGAによる高速ラベリングのパフォーマンスについて述べる。さいごにラインセンサカメラへの実装・実験計画について説明し生産技術分野での応用可能性を述べる。

1. 緒言

近年の画像処理技術の進化とともに、ハードウェアに進化もめざましいものがあり、特にFPGAを用いたライン処理と汎用プロセッサを用いたフレーム処理を併用するシステムが数多く提案されている。また、カメラの高速化から大量の画像データを処理する機構・手法の開発が求められている。特に、ロボットビジョンの分野においても高速な処理系の設計は重要であり、さまざまなアプローチが試されている。そこで我々は、代表的な高速アルゴリズムである走査線アルゴリズムをFPGA化することで高速化を行ってきた。走査線アルゴリズムは2値化されたデータをラン長データ化し、これの結合部分を見つけることでデータのラベル化を行うアルゴリズムであり、画像のライン処理をリアルタイムで行えるFPGAにとって高速化しやすい要素を含んでいる。

本稿では、まずFPGAを用いた高速画像処理システムについて述べる。次に今回対象とするモフォロジーフィルタとそのパラメータ設計について述べる。さらにこれを最適化問題としてとらえ、進化計算での解決を試みた実験とその結果について述べる。また、FPGAを用いたロボットビジョン分野での高速ラベリングシステムの実例を紹介し、プリント基板検査用の高速ラベリングシステムの実現性について検証する。

2. FPGAとリアルタイム画像処理システム

FPGAでの処理はリアルタイムに処理でき、高速である利点がある。ところがFPGAでの画像処理は独特な処理方法であり、画像処理の知識[1][2]とFPGAの開発言語の知識を必要とする[9][10][11]。また、実装されている基板の特性を理解できていないとパフォーマンスを引き出せない状況であり、これらを解決するためのアプリケーションツールの開発は急務とされている。さらに、近年の画像処理パッケージソフトは進化を続けておりこれらとの親和性を捕らえたソフトウエアデザインも求められている。本稿ではモフォロジー処理（Morphological Operations）[1][2]に着目し、このフィルタデザインを進化計算で自動設計する手法を提案する。また、これらをC言語とOpenCV[3][5][6]を用いてPCに実装し画像処理実験を行った。デザインされた処理はVerilogHDLで記述できることが検証出来ているためオフラインでの設計を容易にする。ターゲットとなる

画像処理ボードの FPGA モジュールデザインに準拠する形式で検証した。今回の開発では、モフォロジー処理に限定している。これは進化計算を行う際の設定が容易であるためである。また、VerilogHDL[10] での記述も容易であることも付け加えておく。進化計算を行う場合、処理の仕組みを遺伝子構造にコーディングする必要がある。そこで、今回は 2 値化処理後のバイナリーデータを論理演算するプロセスのみを扱い、これらの処理手順やフィルタのパラメータを遺伝子にコーディングする。適応度関数には、グレー画像を用いる。基板欠陥のグレー画像と欠陥の無いグレー画像を用意し、これらを比較評価とする関数を定義し、これを適応度関数としてあつかう。

FPGA による画像処理を構築するにあたり、㈱ユーテック製 [4] の画像処理ボード gTOP シリーズをターゲットとした（以下 gTOP）。この画像処理ボードは PCI Express で PC に接続し様々なカメラに対応している。 カメラからの信号入力を FPGA で処理後に PC 転送できるようになっている。

gTOP-E1/CLP3TR64M（Altera 製 Cyclone EP 1C12Q 240C8）の外観を図 1 に示す。 FPGA での処理を VerilogHDL で記述することが可能であり、高速な処理が期待できる。既に gTOP の FPGA 用にモフォロジィー処理モジュールがある。モフォロジー処理は膨張処理・収縮処理を複数回組み合わせによって目的の画像を得る。本研究では、この回数組み合わせを組み合わせ最適化問題として扱い、自動設計する手法を実験した。（図 2 参照）

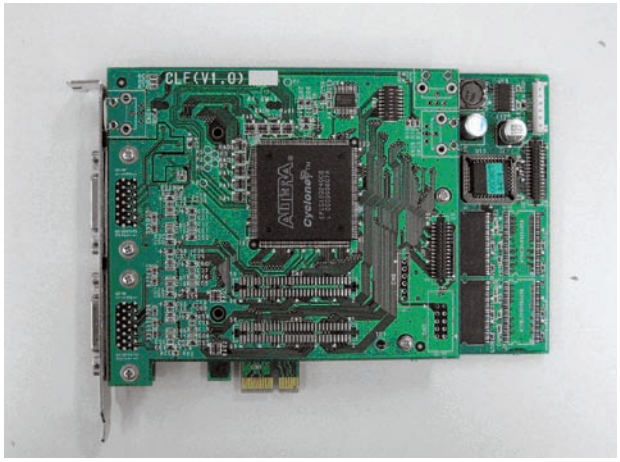

図 1　FPGA 付き画像処理ボード

3. 実験手順と処理の流れ

今回ターゲットとする画像処理ボードは、FPGA での処理を VerilogHDL で記述できる。VerilogHDL の処理開発には専門的な知識が必要であり、また gTOP 内部のバス構造を理解する必要がある。そこで gTOP に実装されているモジュールを参考に、図 2 に示すような処理系を PC 上に作成し、自動設計プログラムで実際の gTOP 上の処理手順と同じように PC でシュミレーションする。自動設計には OpenCV と SI-3[7] プロジェクトの処理サンプルを用いる。VerilogHDL のシミュレーションには ModelSim を用いて実際の処理内容は事前に確認している。また、iverilog コンパイラを併用すれば cygwin 上で動作する。従って PC の実行形式で波形の処理シミュレーションが可能である。

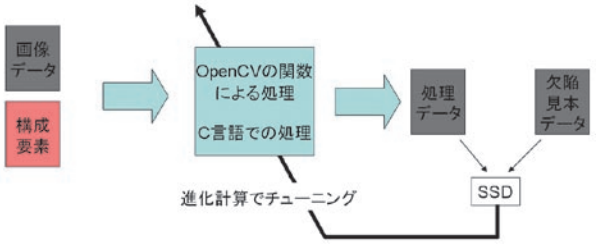

図 2　処理全体の流れ

4. モフォロジー演算器の制御レジスタ

gTOP の FPGA に実装されているモフォロジー演算器の制御レジスタ内容は、以下の通りである。このレジスタで行う処理を PC 上に実装し進化計算により各パラメータを設計していく。bit0 ～ bit19 はモフォロジー演算器の制御レジスタとなっている（図 3 参照）。bit20 ～ bit23 は処理後の画像に対するメディアンフィルタのコントロールレジスタとなっており、0 の場合はメディアンフィルタ処理を行わない。

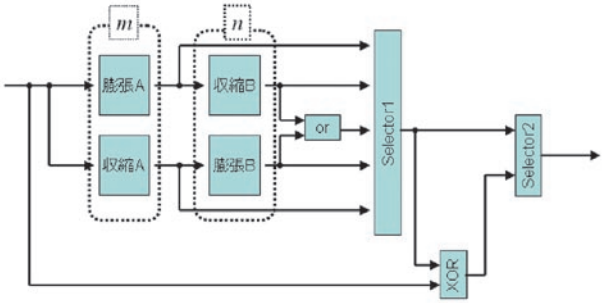

図 3　実装されているモフォロジー演算器
（レジスタ bit0 ～ bit19 でコントロール）

また、bit24～bit31 はモフォロジー演算器に入れるデータを前処理するレジスタであり、bit24 がネガポジ処理の on-off であり、bit25～bit31 が 2 値化の閾値となっている。

表 1　モフォロジー演算器の制御レジスタ
（レジスタ bit0～bit19 でコントロール）

bit0	4 or 8 近傍選択
bit1～bit3	Selector1 （膨張，収縮，Closing, Opening, Closing, Opening と Closing の OR, Opening と Closing の XOR）
bit4	Selector2 （入力画像と Selector1 処理画像との XOR）
bit8～bit11	パラメータ m（3x3～13x13）
bit12～bit15	未使用（実ボードのみで使用）
bit16～bit19	パラメータ n（3x3～13x13）
bit20～bit23	メディアンフィルタ（on-off,3x3～11x11）
bit24	元画像をネガポジ処理
bit25～bit31	2 値化閾値

3. 進化計算による制御レジスタの最適化

遺伝的アルゴリズム（GA : Genetic Algorithm）のスケジューリング問題への適用と題して本研究は行われている。スケジューリング問題とは最適化問題と呼ばれる問題の 1 つで、最適化問題には他にもナップサック問題や、巡回セールスマン問題、エイトクイーン問題等が含まれる。このような問題は、人の手で解くには困難であり、多大な時間が必要である。従ってこれらはコンピュータを用いて解く方法が一般的であり、その解き方も複数ある。そして、その解き方の 1 つに GA がある。画像処理分野に多くの適用例がある。[12][13]

GA とは遺伝子で表現した「個体」を複数用意し、適応度（fitness）の高い個体を優先して選択して交叉（組み換え）・突然変異等の操作を繰り返しながら解を探索するものである。適応度は適応度関数によって与えられる。目標とする適応度（fitness）には処理結果の画像とトレーニング用画像を比較する SSD# を定義する。

適応度関数 $SSD^{\#}$ の定義

全画素に対して演算する。ここで、処理結果画素 $= r(x,y)$、トレーニング画素 $= t(x,y)$ とする。画素比較得点関数 z を次のように定義する。

if$(r(x,y), t(x,y))$ is（白、白）then $z(x,y)$　$= 100$
if$(r(x,y), t(x,y))$ is（黒、黒）then $z(x,y)$　$= 10$
if$(r(x,y), t(x,y))$ is（白、黒）then $z(x,y)$　$= 1$
if$(r(x,y), t(x,y))$ is（黒、白）then $z(x,y)$　$= 0$

次に制御レジスタの各 bit の値を評価する *num_com* を次のように定義する。

num_com = 制御レジスタ中の 0 の数

これはレジスタ内に 0 が多いほど大きい値をとる。これにより実際には処理に影響しない bit を 0 にする作用を持つ。最終的に定義した関数を次に示す。

$$SSD^{\#} = \left(\sum_{y=1}^{HighWigth} \sum_{x=1} z(x,y) \right) \div (High \times Width) \times 10000 + num\_corm$$

num_com を画素比較値に加えることで、トレーニング画像に対して同じ評価であっても、0 の多い方が bit12～bit15 のように情緒な bit 情報を省くことが出来る。遺伝演算では $SSD^{\#}$ を適応度関数 fitness 関数として用いる。ペアリングでは、適応度の高い順番に 2 つの個体をランダムに選びペアとする操作を行う。ただし、エリート保存を行うため、エリートを選択しないこととした。交叉には 1 点交叉をもちいた . 突然変異は生物に見られる遺伝子の突然変異をモデル化したもので、個体の遺伝子の一部を変化させる操作である。今回は一定の確率でレジスタの 0 と 1 を反転している。遺伝演算のパラメータは以下のとおりである。

1. 遺伝子：制御レジスタの bit 幅であり 1 個体を形成
2. 20 個体を集団とする
3. 1 個体につき 1 つの $SSD^{\#}$ 評価を適応度として持つ
4. 交叉確率は 50% 突然変異確率 5%
5. 打ち切り世代 20 万世代

4. シミュレーション実験

実験環境は Pentiumu4 3GHz の PC を用い WindowXP 上の Cygwin に作成した。コンパイラは gcc、g++ を用いた。テスト用画像には精密工学会画像応用技術専門委員会が開催している外観検査アルゴリズムコンテストの画像データ 2007 年度版から 1 枚を用いた [8]。トレーニング画像

はペイントソフトで図 6 のように加工して作成した。

実験 1 .

前処理の画像反転＋ 2 値化を人の手によって画像を見ながら行い。そのパラメータ部分（bit24～bit31）を（c5）[16]（ネガポジ処理 on、閾値 196 で 2 値化処理）に固定し、残りレジスタを探索させた。つまり、図 5 の元画像に対し図 7 のような処理を行った画像をモフォロジー演算器に入力することになる。（図 4 参照）探索は初期収束を起こし、25 世代（4 分）で図 8 のような処理結果を得る制御レジスタパラメータ（530008）[16] を得ている。適応度計算に用いたトレーニング画像は図 6 を示す。この処理は、4 近傍の Opening 処理であり、3x3 収縮処理後に 13x13 膨張処理を行い、最後に 7x7 メディアンフィルタ処理を行っている。若干の誤検出部分もあるが、全体としてゴマノイズも少なく、キレイな検出画像となっている。（図 8 参照）

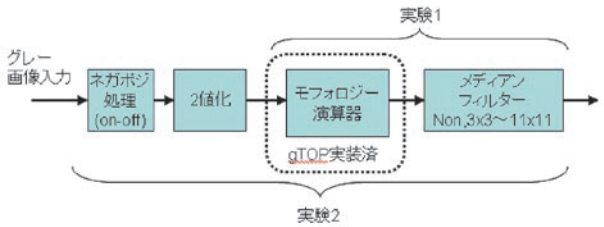

図 4　前処理＋モフォロジー演算器＋後処理

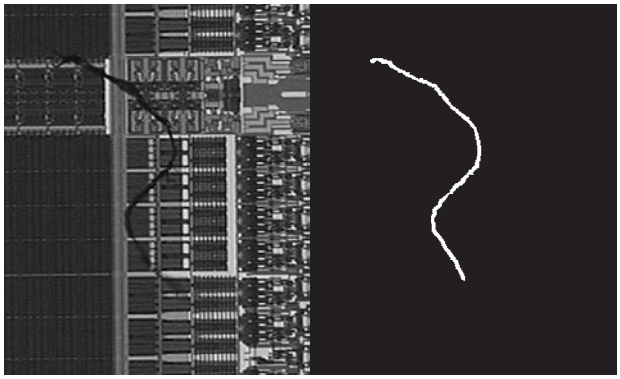

図 5 原画像

図 6 トレーニング画像

実験 2.

前処理の画像反転＋ 2 値化も含めて進化計算でパラメータを探索した。全 bit（bit0～bit31）を探索させた。探索は局所解に捕まり、500 世代（64 分）で図 10 のような処理結果を得る制御レジスタパラメータ（3e20031a）[16] を得ている。適応度計算に用いたトレーニング画像は図 6 を示す。この処理は、閾値 62 で 2 値化後モフォロジー演算器に入力している。モフォロジー演算では、4 近傍の Opening 処理と Closing 処理の OR 処理であり、9x9 収縮・膨張処理後に 3x3 膨張・収縮処理を行い、最後に 5x5 メディアンフィルタ処理を行っている。この制御レジスタの処理は、図 5 の元画像に対し図 9 のような処理を行った画像をモフォロジー演算器に入力することになる。（図 3 参照）若干の誤検出部分もあり、全体としてゴマノイズもあるが、形状をはっきりと検出している画像となっている。（図 10 参照）　これらの結果から進化計算によって異物検出処理を自動的に作り込むことが出来ることがわかった。今後はさまざまな条件での実験が必要である。

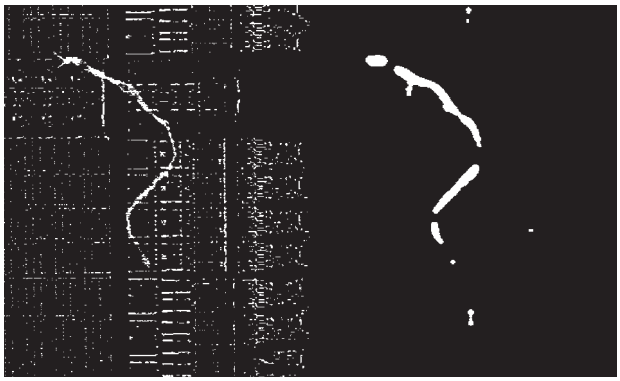

図 7　ネガポジ＋ 2 値化（単体）　図 8　処理結果（単体）

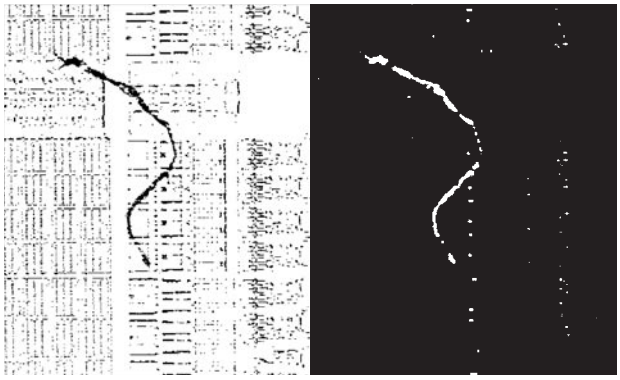

図 9　2 値化のみ（同時）　図 10　処理結果（同時）

4. FPGA による高速カラーラベリング

FPGA によるラベリング処理を構築するため、㈱ユーテック製の画像処理ボード gTOP-E1/CLP3TR 64M を用いた（以下 gTOP）。この画像処理ボード PCI Express で PC に接続しカメラリンク対応となっている .

カメラからの信号入力を FPGA（Altera 製

Cyclone EP1C12Q240C8）で処理後に PC 転送できるようになっている。gTOP の外観を図 11 に示す。FPGA での処理を VerilogHDL で記述することが可能であり、高速な処理が期待できる。

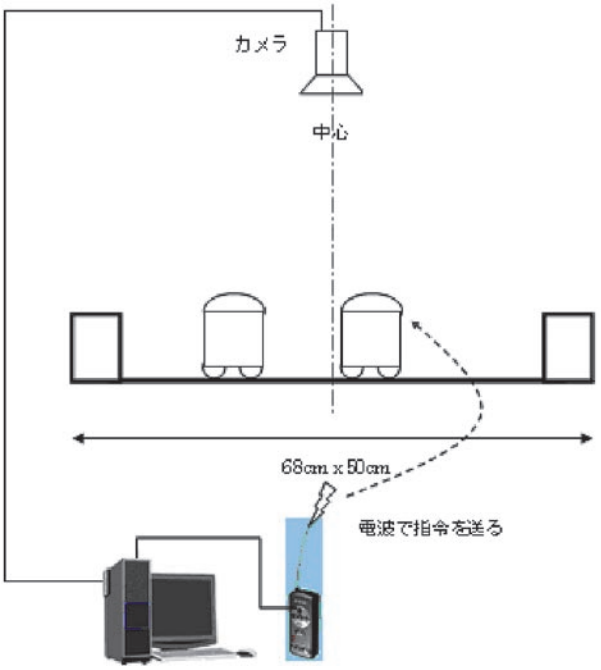

図 11　カラーラベリング実験環境

図 12　JAI 製 CV-M71CL

カメラには JAI 製 CV-M71CL を用いた。CV-M71CL は 46 万画素 CCD（782H × 582V）を使ったデジタルカラープログレッシブスキャンカメラで各 8 ビットの RGB カラーデータ信号をカメラリンクケーブルから得ることができる。フレームレートは 60fps・37MHz で転送できる。

今回の実験ではロボットビジョン分野を想定し、サッカー用ロボットを題材としてラベリング実験を行った。ブレイブ社の Mr. SOCCER を用いる。図 13 に Mr. SOCCER の外観をしめす。本研究ではボールを高速追跡するためボールの色でカラーラベリングをおこなう。

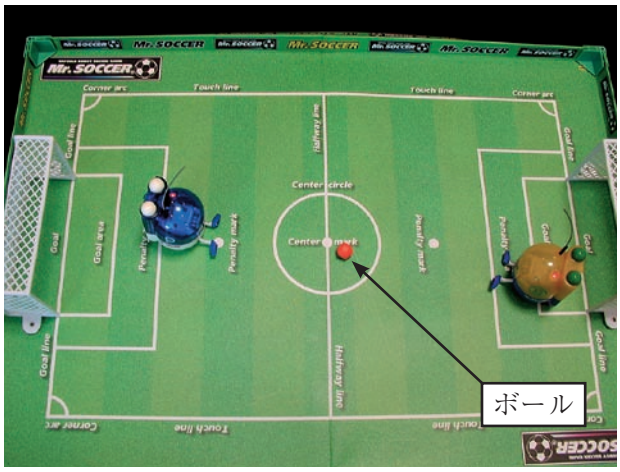

図 13　ブレイブ社の Mr. SOCCER

用いるフィールドは、68cm x 50cm あり、これを中央部分から頭上 5m 程度上から CCD カメラを使って画像を取得する。高速にボールを追跡する実験を行った。gTOP を用いて取得した画像からボールをラベリング処理で見つける処理を構築した。得た情報から位置・向き・速度を算出することが出来る。ボールはオレンジ色をしているため、HSL 色抽出・高速カラーラベリングを行う。最初に取得画像の RGB 値を HSL データに変換し 2 値化する。オレンジの部分を取り出すため閾値は次のとおりである。

H:200～360 度 0～130 度 L:50～150 S:10～110

HLS を用いて色相基準で取り出すことにより、RGB で行うより照明からの明るさの違いに影響されにくいシステムを構築できる。ロボットサッカーを扱う環境下では、工場などとは違い照明が安定的に得られないことが多い。そのことを考慮すると HLS は有効に働くと考えられる。これらの閾値で分けた 2 値情報から走査線アルゴリズムを用い、ラン長を作る。ラン長はランレングス法や連長法とも呼ばれ、画像圧縮アルゴリズムによく用いられる。一般に白色と黒色のみで構成された 2 値画像で隣り合った画素の色の変化が少なく、値が連続して出現するデータの冗長性を利用し、「同じ色が連続する長さ」を符号化する手法である。

本研究では、このラン長集合を結合することでラベリングを実現している。このラベリング手法の特徴は、ラン長の結合をすると同時に面積、*x,y* の最大値、*x,y* の最小値を計算することが出来る点がある。

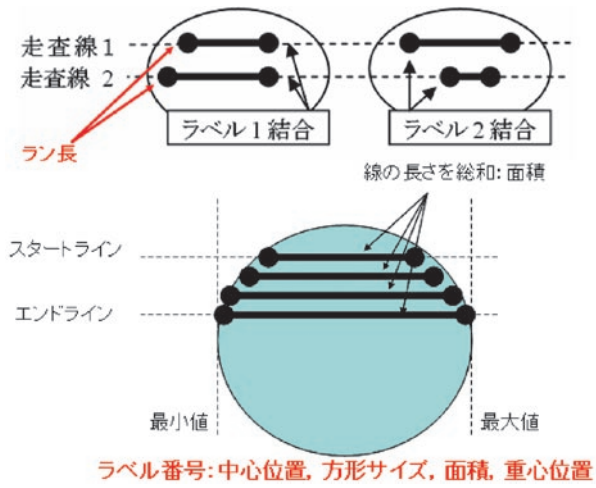

図 14　走査線アルゴリズム

面積は、各ラン長を結合する時に横の長さを足し算していけば最終的な結果が得られる。また、x 座標の最小値・最大値はラン長の出発点と最終地点の x 座標を比較し更新すればよい。また y 座標の最大値は結合が決まった時に +1 すればよい。y 座標の最小値は最初の y 座標となる。但し、Y 字型や U 字型の画像を結合するときにのみ更新が必要となる。これらの計算はラン長取得する度に計算することができるため、方形走査をせずにライン走査のみでラベリングを行うことが可能であり、非常に効率の良いアルゴリズムとなっている [14][15][16][17][18]。

これらの処理は PC 上で行った。実験用 PC には WindowsXP+cygwin、IntelCore2Duo 2.66GHz、2GB メモリの PC を用いた。画像処理ライブラリには OpenCV を用い cygwin 上 gcc を用いて開発・実験を行った。

OpenCV[1] はインテルが開発・公開しているオープンソースのコンピュータビジョン向けライブラリである。[6][7] プラットフォームとして Windows ならびに Linux、FreeBSD、Mac OS X 等をサポートしている。C/C++ で記述されており、Windows の場合 DLL をインポートすれば Cygwin や VisualC/C++, VisualBasic などでも使うことが出来る。

実装分野は数多くあり、フィルターなどの一般的な画像処理に加え、機械学習、サポートベクターマシン、決定木などのアルゴリズムもサポートしている。本研究では、gTOP で取得した画像を OpenCV で処理するため、SI-3 プロジェクトで公開されている技術を用いて Cygwin（g++）での開発を行った。[8]

図 15　ラベリング結果

```
                :
num_of_pixels: 186
center: 557,482.5
size:   17,16
min:    549,475
max:    565,490
center_of_graivty: 556.866,482.774
source_value: 255
result: 57
                :
```

図 16　処理画面

実験では 10 個のボールをフィールドにばら撒き、gTOP で取得した画像を処理した。一回の撮像で 70 個程度の候補は選出された。各データは、図 16 のように画面に出力される。

ボールを判定するため図 15 に示すように 150 ピクセル以上のラベルを取り出して十字でマーカーをつけた。これらの処理を PC のみで処理した場合、ラベリング時間は 31.0ms であった。取り込み・処理・表示時間を含めると全体の処理時間は 124.5ms であった。ソフトで処理する場合ワークメモリへの処理・転送がかなり発生する。このため時間が掛かる。そこで、ラン長を作るための 2 値データを FPGA で処理し PC の処理を軽くすることで高速化を試みる。

5. FPGA による高速カラーラベリング

gTOP には FPGA オプションボードがついており、これらを VerilogHDL でカスタマイズできる。今回のモジュールは「Quartus-II8.0sp1 Web Edition」と「ModelSim-Altera 6.1g Web Edition」を使って開

発した。[10][11][12][13]　開発した処理を図 17 に示す。カメラから入ってきた RGB を HLS 変換モジュールに入れる。モジュールから出力された HLS 値から判別モジュールで 2 値情報にし、その結果を元画素の RGB の B（青）の bit0 に入れて画像データとして PC に転送した。各画素の B の bit0 はラン長を作るための 2 値データとなるため、PC 側で結合することでラベリングを行っている。

PC 側では取得した画像の画素のうち青（B）の bit0 のみを見て行き、0 → 1 もしくは 1 → 0 の変化する座標からラン長を計算する。

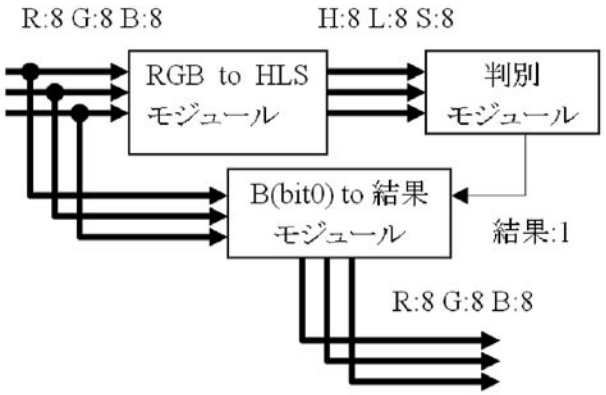

図 17　高速カラー処理モジュール

比較実験を行うため同じ環境で処理時間を計測した。FPGA で処理する場合、PC 側のアルゴリズムは若干変更となる。HLS への変換と 2 値化処理がなくなりその部分を省いてその他の処理はそのまま PC で行った。その結果、ラベリング時間は 7.0ms となった。カメラからの取り込み・処理・表示時間を含めた全体の処理の場合、処理時間は 93.5ms であった。PC がラベリングに要した時間は 24.0ms（77%）改善され、全体の処理としては 31.0ms（25%）改善された。FPGA での処理が注目画素から 19 クロック（513ns）送れて処理が終わる。この時間を考慮しても十分な改善が見られている。

ラベリング時間：31.0ms　→　7.0ms
全体処理時間：　124.5ms　→　93.5ms

6. 高速ラベリングへの応用可能性について

現在では、産業システムとしてのフィールド実験を考えている。ファクトリーオートメーションでの異物検査にはさまざまな分野と工程があり、現在では*デジタルラインセサカメラ*を使って一度にデータを取り込み、巨大画像（数万×数万）からの異物検出実験を行っている。そのために準備している。ラインカメラの場合は水平画素が数千～数万画素になるため、水平同期時間が FPGA の処理時間に対して十分に長い。つまり、水平をスキャンする時間は数百ミリ秒単位であるため、次のラインをスキャンしている時間の間に一本前のデータ処理で処理できる。したがってリアルタイム処理がやり易く、リアルタイムカラーラベリングのもっとも活躍できる分野である。現在では、図 18 に示すような簡易的な装置を使って実験を行っている。これは図 19 に示すような工場での検査システムを想定している。

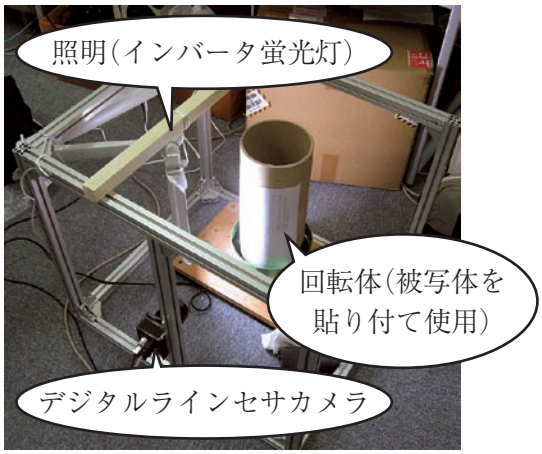

図 18　簡易システム

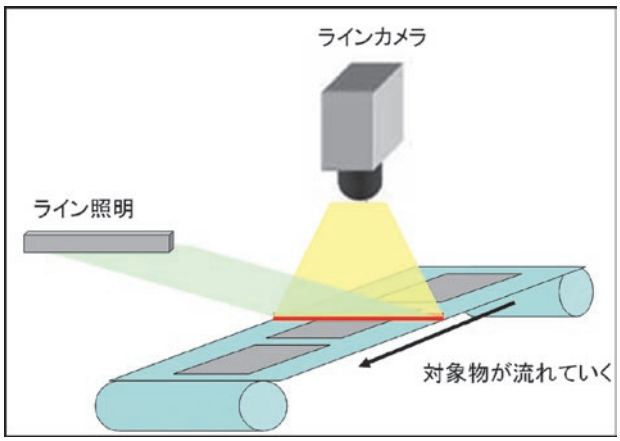

図 19　異物検査システム

7. 結言

現在、デジタルラインカメラを使って簡易のシステムで開発を行っている。これからは外資金を調達し、十分な予算で実現場に近いシステムの構築をめざす。実際に近い環境でのデータ収集とソフト開発が目的となる。現在、これらのデータと

成果を学会発表や論文投稿で対外に発表するように準備している。しかし、これだけにはとどまらず、大学機関であることを活かし画像処理エンジニア育成のためのセミナーなどを開催し、生産現場へのフィードバックを掛けたいと考えている。さらに地方産業の技術連携も考えており、そのためのセミナー開催や設備投資は学術支援の一環として妥当であり、必要であると考えている。

文献

[1] 井上 誠喜、林 正樹、三谷 公二、八木 伸行、中須 英輔、奥井 誠人：C 言語で学ぶ実践画像処理、オーム社（1999）

[2] 岡崎 彰夫 : はじめての画像処理技術、工業調査会（2000）

[3] 奈良先端科学技術大学院大学 OpenCV プログラミングブック制作チーム：OpenCV プログラミングブック、毎日コミュニケーションズ（2007）

[4] 株式会社ユーテック：
http://www.utech-corp.co.jp/

[5] Intel Open Source Computer Vision Library：
http://www.intel.com/technology/index.htm

[6] OpenCV：http://opencv.jp/

[7] SI-3 プロジェクト
http://www.utech-corp.co.jp/si3/index.html

[8] 外観検査アルゴリズムコンテスト 2007 : 精密工学会画像応用技術専門委員会
http://www.tc-iaip.org/alcon2007/

[9] 日本アルテラ： トレーニングコース Quartus II 基礎編 テキスト

[10] 小林 優：改訂 入門 Verilog HDL 記述、CQ 出版社

[11] 堀 桂太郎：図解 ModelSim 実習、森下出版株式会社

[12] 長尾 智晴：進化的画像処理、昭晃堂（2002）

[13] 白川 真一、長尾 智晴：Graph Structured Program Evolution への自動関数定義の導入とその評価、進化計算シンポジウム 2008

[14] 何 立風、巣 宇燕、鈴木 賢治、中村 剛士、伊藤 英則 , 3 次元 2 値画像における高速ラベル付けアルゴリズム , 電子情報通信学会論文誌 , Vol.J92-D,No.12,pp.2261-2269, 2009.

[15] 何 立風、巣 宇燕、鈴木 賢治、中村 剛士、伊藤 英則、同等ラベル解析に基づく 1 回走査ラベル付けアルゴリズム、情報処理学会論文誌 , Vol.50, No.6, pp.1660-1667, 2009.

[16] 何 立風、巣 宇燕、鈴木 賢治、中村 剛士、伊藤 英則、ラスタ走査型ラベル付けアルゴリズムにおける第一走査の効率化手法、電子情報通信学会論文誌 , Vol.J92-D, No.6, pp.951-955, 2009.

[17] 何 立風、巣 宇燕、鈴木 賢治、中村 剛士、伊藤 英則、連に基づく高速 2 回走査ラベル付けアルゴリズム、映像情報メディア学会誌、Vol.62, No.9, pp.1461-1465, 2008 年 9 月号 .

[18] 何 立風、巣 宇燕、鈴木 賢治、中村 剛士、伊藤 英則、高速 2 回走査ラベル付けアルゴリズム、電子情報通信学会論文誌、Vol. J91-D, No.4, pp.1016-1024, 2008.

[19] 星野孝総：進化計算によるモルフォロジー演算の最適設計と FPGA モジュールへの実装、画像センシングシンポジウム 2009（SSII09）、CD-ROM IS1-29 、パシフィコ横浜

[20] 星野孝総：Particle Swarm Optimization を用いた階層型ニューラルネットワークの学習と性能の検証、進化計算シンポジウム 2008、平成 20 年 12 月 20-21 日、北海道 登別温泉 ホテル まほろば

[21] 星野孝総：FPGA を用いた高速ラベリングによるロボットサッカーボール追跡と中継システムの提案、ViEW2008 ビジョン技術の実利用ワークショップ 2008/12/4-5、CD-ROM I-46、パシフィコ横浜

The report of an application possibility about high speed labeling process using FPGA and an automatically optimization method using evolutional computation for Morphological Operations

Yukinobu HOSHINO

（Received : May 6th, 2010）

*School of Engineering, Kochi University of Technology
185 Miyanokuchi, Tosayamada, Kami city, Kochi 782-8502

E-mail: hoshino.yukinobu@kochi-tech.ac.jp

Abstract: In this paper, we report that we think of the basic inspection which used an image processing system and making system hardware about the possibility of the application through the implementing experiment. This system takes up and it confirms the example of FPGA. In recent years, a lot of practical example is already reported too much, and is fixing the high-speed real-time processing which used FPGA at present as the circuit board technology which supports manufacturing technology. Also, this system is used at the production site in a lot of companies. As the education of the engineer, the university must educate those FPGA technologies. The engineers have to design the image processing hardware by the computer program and the technical knowledge about the image processing design. To support these, we aimed at the system research. Therefore, using evolutional computation is new approach about computing of the best control parameter as offline. And we are explained and moreover the performance of the high-speed label bill which depends on FPGA is described. At last, we discuss the possibility of the application of the manufacturing technique field using our technique and the linear sensor camera for huge digital image process.