コンピュータビジョンと
イメージメディア 136－18
（２００３．１．１７）

ウェーブレット変換を用いたリアル タイム追跡回路

木塲　俊暁† 吉井　圭吾‡ 伊藤　光‡ 工藤　健慈‡

今中　晴記‡ 上野　貴史‡ 志賀　裕介‡ 金丸　隆志‡ 関根　優年‡

†東京農工大学
〒184-8588 東京都小金井市中町 2-24-16
E-mail: †t-koba@sekine-lab.ei.tuat.ac.jp, ‡sekinem@cc.tuat.ac.jp,
‡{k-yoshii,hikaru,k2x,imanaka,ueno,you,kanamaru}@sekine-lab.ei.tuat.ac.jp

あらまし 本論文では Haar ウェーブレット変換を用いた追跡回路を提案する。処理の高速化を計る為、次の２点を工夫した。１つは、圧縮画像を用いてテンプレートマッチングすることである。これによりデータ量が減る為、演算量を減らすことができる。もう１つは､対象物の位置から予測領域を決定し、これを３つの領域に分け同時にテンプレートマッチングすることである。これにより探索領域が狭まる為、演算量を減らすことができる。本研究では、予測領域をテンプレートマッチングするのにかかる時間をシミュレーションから計算した。

キーワード 多重解像度，ウェーブレット変換，テンプレートマッチング，FPGA，

A real-time tracking circuit using Wavelet Transform

Toshiaki KOBA†, Keigo YOSHII, Hikaru ITOH‡, Kenji KUDOH, Haruki IMANAKA,

Takashi UENO, Yuusuke SHIGA, Takashi KANAMARU and Masatoshi SEKINE‡

†Tokyo University of Agriculture and Technology
Nakamachi 2-24-16, Koganei-shi, Tokyo, 184-8588 Japan
E-mail: †t-koba@sekine-lab.ei.tuat.ac.jp, ‡sekinem@cc.tuat.ac.jp,
‡{k-yoshii,hikaru,k2x,imanaka,ueno,you,kanamaru}@sekine-lab.ei.tuat.ac.jp

Abstract A tracking circuit by using Haar wavelet transform is proposed. In order to reduce processing time, the following two features are included: First, a template matching proceeds on compressed input images for the reduction of operations. Next, a search prediction from the current target position is designed with dividing the search domain into three partial domains, where concurrent template matching operations go on. This reduces the execution time of operations. In this paper, I present the execution time of the template matching in the prediction domain by estimation from a logic simulation of the circuit designed for a FPGA device.

Key words Multi-resolution, Wavelet transform, template matching, FPGA

1．はじめに

動画に対する追跡処理の研究は数多くなされてきた。その中で、テンプレートマッチングを用いて追跡するものがある。テンプレートマッチングでは、予め撮像された画像（テンプレート）の位置を相関演算を用いて判定する手法で、この相関演算は単純な計算の繰り返しで行われる。

本研究では処理の高速化を計る為、coarse?to?fine テンプレートマッチングを用いる。coarse-to-fine テンプレートマッチングでは、圧縮画像を用いて対象物の場所決めを行う[1][2]。しかし、この手法はコーステンプレートで得た場所が必ずしもファインテンプレートで得られる場所と一致しないという問題点がある。そこで、コーステンプレートにより対象物の候補を複数見つけ、その中からファインテンプレートで一致度の高いものを選ぶ方法が考えられる。

本研究では、追跡対象物の予測領域を決め、その領域でテンプレートマッチングを行う。これは、MPEG の動きベクトル検出用 LSI と似ているが、回路の構造は異なっている[3]。動きベクトル検出用 LSI では､比較計算をする部分がシストリックアレイ形（同じ回路を複数並べたもの）で回路規模が大きくなるが、本研究では、回路規模を抑えるために１つの回路で実現している。

この回路は、本研究室で試作した FPGA ボード HwModule(図 1)に実装する。

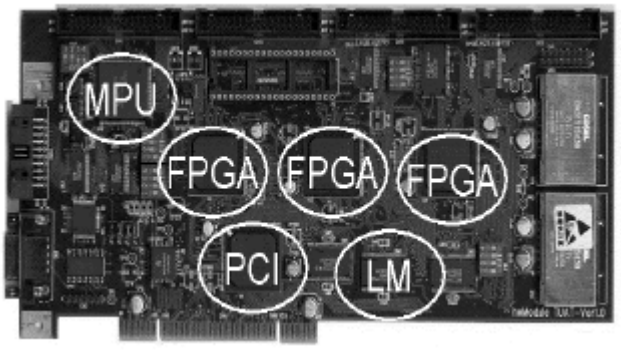


図 1: 本研究室で試作した HwModule

2．追跡処理手法

2-1．Haar のウェーブレット変換

本研究においては、追跡処理に Haar 関数による離散ウェーブレット変換された画像を使用する。Haar のスケーリング関数 $f_H(x)$ と Haar のウェーブレット関数 $y_H(x)$ は式(1)(2)のように定義される。

$$f_H(x) = \begin{cases} 1,\ 0 \leqq x < 1 \\ \\ 0,\ \text{otherwise.} \end{cases} \tag{1}$$

$$y_H(x) = \begin{cases} 1,\ 0 \leqq x < 1/2 \\ -1,\ 1/2 \leqq x < 1 \\ 0,\ \text{otherwise.} \end{cases} \tag{2}$$

スケーリング関数 $f_H(x)$ を Lowpass Filter、ウェーブレット関数 $y_H(x)$ を Highpass Filter として入力画像（Original image）の水平方向・垂直方向の順に１次元変換を行うことで 1/4 に圧縮された低周波成分（Scaling image)と高周波成分(Wavelet image)が得られる(図 2)。

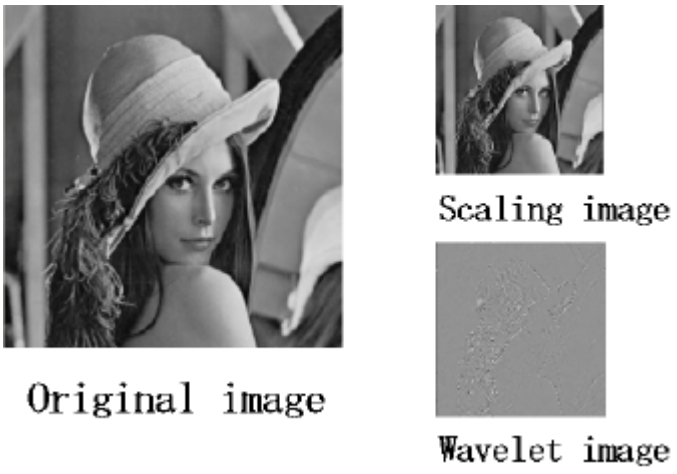


図 2 ： Haar-Wavelet 変換による
平滑化とエッジ抽出

2-2．多重解像度

本研究では、Haar 関数による離散ウェーブレット変換の多重解像度表現を利用した追跡処理をする。

図 3 のように、ウェーブレット変換を１回行う

ことを１レベル変換とし、ここで得られる元画像を 1/4 に圧縮した Scaling image と Wavelet image をレベル１とする。レベル１の Scaling image をさらに１レベル変換することで、元画像を 1/16 に圧縮したレベル 2 の Scaling image と Wavelet image が得られる。このように、ウェーブレット変換してできた低周波成分をさらに変換するという作業を繰り返すことにより、様々な圧縮レベルのデータを得ることができる（図 3)。これを多重解像度という。

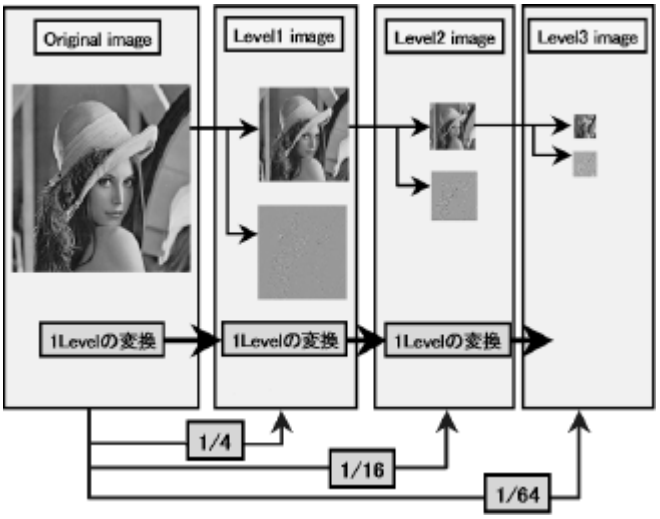


図 3 ： 多重解像度

2-3．追跡手法

本研究では、入力画像の追跡対象物をターゲット、そのターゲットの比較データとなるものをテンプレートと定義する。テンプレートはウェーブレット変換で圧縮し、各レベル毎の解像度でテンプレートを持たせて、多重解像度表現している。画像センサからの入力画像も同様に圧縮し、この画像と同レベルに圧縮されたテンプレートをテンプレートマッチングすることで、ターゲットが移動した位置を求める。

3．処理の流れ

まず、**CMOS sensor** から原画像(256×256)を **Wavelet chip** に取り込む。次に **Wavelet chip** で、取り込まれた画像をウェーブレット変換し、この変換結果を Host と Tracking circuit に出力する。

Host では、ウェーブレット変換した入力画像をテンプレートマッチングすることでターゲットの座標を決定する。座標が決定したら、Tracking circuit に対しターゲットの座標とテンプレートを送る。同時に、Wavelet chip に対し Tracking circuit にデータ出力を行うように制御信号を送る。

Tracking circuit は、Host から受け取ったターゲットの座標からターゲットが何処にあるのかを Wavelet chip から送られた圧縮画像に対し予測演算を行い、追跡する。

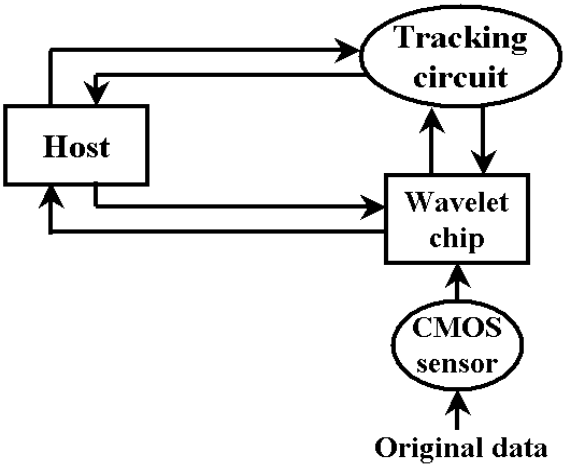


図 4 ： 処理の流れ

4．Tracking circuit

4.1 Tracking circuit の動作

本研究において、予測演算を行うために入力画像から切り出された領域を予測領域とする。これにより入力画像全領域ではなく予測領域だけをテンプレートマッチングすることになり、演算量を減らすことができる。また、テンプレートマッチングに圧縮画像を用いる為、更に演算量を減らすことができる。

ここで追跡に失敗した場合、予測領域を変えるか（例えば、横長にしたり、縦長にしたり、拡大したりする)、Wavelet chip に対しコマンドを出して入力画像の圧縮レベルを変えて、再度追跡を行う。追跡に成功した場合は、Host に対してターゲットの移動後の座標とテンプレートマッチングで得られた一致度を出力する。

以下の図 5 は、tracking_circuit の具体的な動

作例である。

※　入力画像は、縦＝8，横＝8 の 64 領域に区切り、各領域に左上から右下まで横方向に 0～63 まで順番を決めている。ここで、1 領域のアドレス数は、Level1 で 16(縦)×4(横)、Level2 で 8×2、Level3 で 4×1 となる。

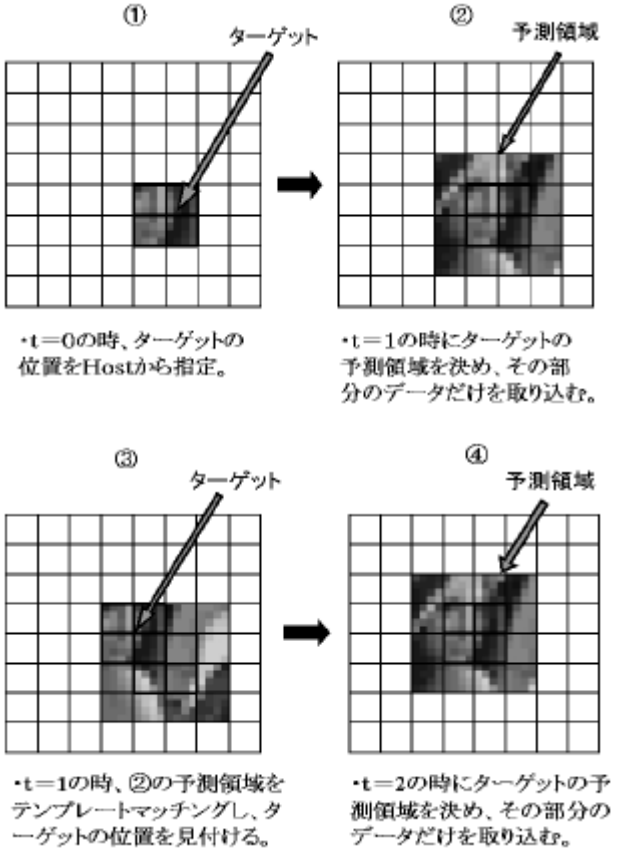


図 5 ： Tracking circuit の動作例

4-2．Local Memory 内のデータの割り当て

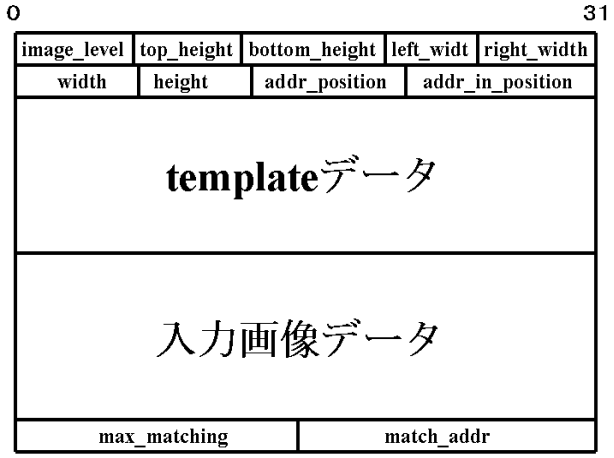


図 6 ： Local Memory 内のデータ

図 6 は、Local Memory 内のデータの割り当てについて示した図である。address0～address1 は、制御用に使用する。address2～address1101 は、テンプレートと入力画像のデータ用に使用する。address1102 は、追跡結果のデータ用に使用する。以下に、address0 と address1、そして address 1102 に入るデータについて説明する。

address 0 には、image_level、top_height、bottom_ height、left_width、right_width の各制御データが置かれる。image_level は、画像の圧縮レベルを表す。top_height、bottom_height、left_width、right_width については、下図 7 に示す。

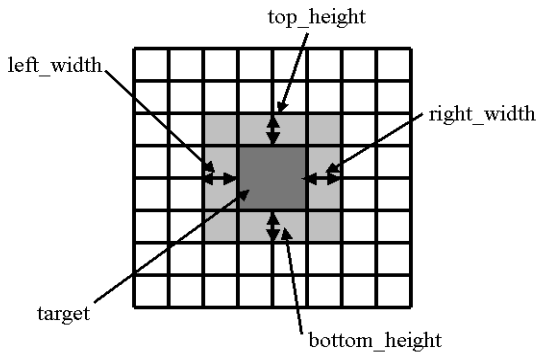


図 7：top_height、bottom_height、left_width、right_width について

※ 図 7 において中央の濃い灰色の部分は target を、濃い灰色と薄い灰色の部分は予測領域を示す。

address 1には、width、height、addr_position、addr_in_position の各制御データが入る。width と height は、それぞれテンプレートの幅と高さを表す。addr_position は、ターゲットの左上隅が位置する場所を指す(address で指定)。addr_in_position については、図 8 を用いて説明する。

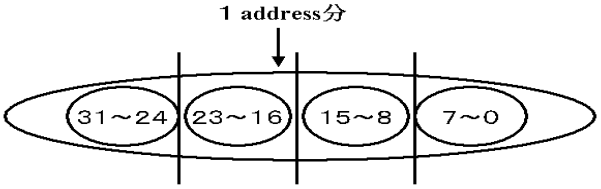


図 8：addr_in_position について

各 address は、1 word 4 field に 4 画素のデータ

をパックする。図 8 で、0～7，8～15，16～23，24～31 にそれぞれ field 番号 0，1，2，3 を割り当てる。ターゲット左上隅の画素がどの field 番号に属しているか示したものが addr_in_position である。

address 1102 には、max_matching、match_addr というデータが入っている。max_matching は、テンプレートマッチングの最小値を示し、match_addr は、テンプレートマッチングが最小値をとる時の入力画像の左上隅の address を示している。

4-3．Tracking circuit のブロック図

以下に、Tracking circuit の各モジュールの説明を図 9 のブロック図を用いて行う。

ⅰ）lm_tr_connect

Local Memory を制御するインターフェースと tracking circuit をつなぐ module。

ⅱ）tg_data_in

addr_position，addr_in_position，そしてテンプレートを読み込む。addr_position と addr_in_position は予測領域を決定する際に用い、テンプレートは Block RAM に格納する。

この Block RAM とは、HwModule に搭載された FPGA の中にあるメモリのことである。

ⅲ）position_rec, rlm_ready, rlm_calc, rlm_rec

tg_data_in で読み込んだ addr_position と addr_in_position から、ターゲットの４隅の位置が６４領域中のどの領域に属するかを求める。

ⅳ）squeeze_ready

ⅲ）で求めた領域を元にターゲットの予測領域を決定する。

ⅴ）part_take_image

予測領域の中から部分的にデータを取り出し、Block RAM に格納する。

ⅵ）template matchig

Block RAM に格納したテンプレートとⅴ）のデータをテンプレートマッチングし、一致度を求める。

1) position_rec によるフィードバック制御

一致度がある閾値より小さい場合、match_addr と max_matching を出力し Local Memory に書き込む。そして、match_addr を元に再度予測領域を決め追跡を行う。

2) squeeze_ready によるフィードバック制御

一致度がある閾値より大きい場合、予測領域を変え、再度追跡を続ける。

（※閾値の値は今後検討していく。）

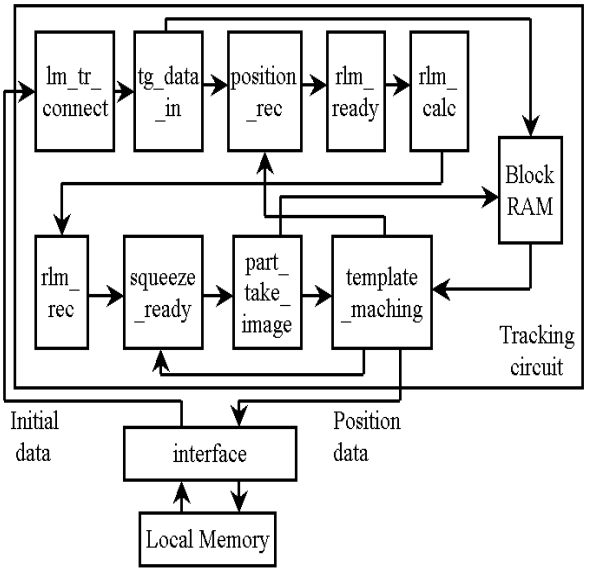


図 9 ： Tracking circuit のブロック図

4-4．Block RAM への予測領域データの書き込み

図 10 は、予測領域の画像に対し列の番号を付けたものである。予測領域データを Block RAM に書き込む時、最初に図 10 の 1 列目から 9 列目の部分を左上から右下まで横方向に書き込む。1 列目から 9 列目のデータの大きさは次のようになる。

横方向：(テンプレートの横幅)＋(1address)

縦方向：{(予測領域の縦幅)-(テンプレートの縦幅)}÷3＋(テンプレートの縦幅)

※Block RAM にこの大きさのデータが入った状態を MAX 状態と本研究で定義する。

図 10 において①の部分からテンプレートマッチングを始めていく。テンプレートマッチングする際、上から下に向かって 1 画素ずつ、ずらしながら行う。①の領域全てテンプレートマッチングが終わったら、次は①の部分を 1 画素分右にずらし、その領域を前と同じ要領でテンプレートマッチングする。これを繰り返し、①の部分が 1 アドレス分ずれた②の部分(左端が 2 列目,右端が 9 列目)に

至った時、10 で示した列のデータを 1 列目のデータが入っていた Block RAM の address に書き込む。その後更に進んで、①の部分が③の部分(左端が 3 列目,右端が 1 列目)に至った時、11 で示した列のデータを 2 列目のデータが入っていた Block RAM の address に書き込む。このようにして、予測領域のデータを Block RAM に書き込んでいく。
(但し、この説明はレベル 1 を例にとったものである。)

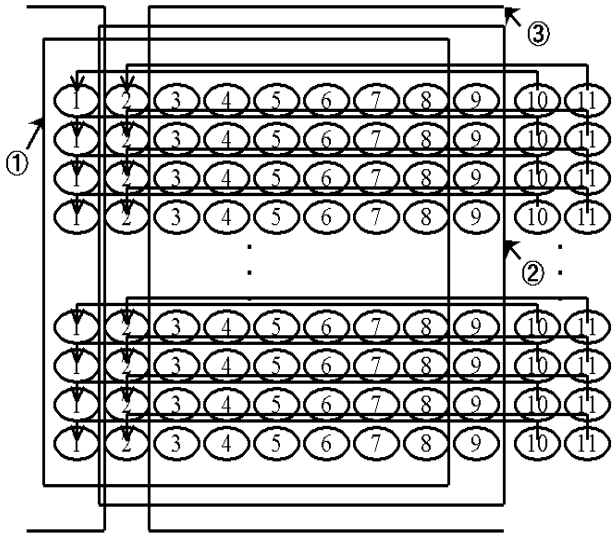
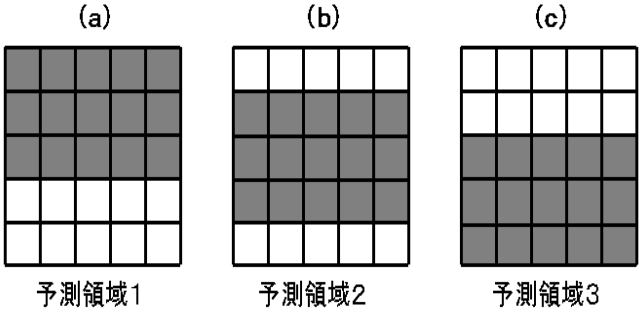


図 10：Block RAM への予測領域
データの書き込み

4-5. Template Matching の並列処理

予測領域に対し 1 つのテンプレートで予測領域全てを Template Matching 計算をすると時間がかかる。そこで本研究では、予測領域のデータを予測領域1～予測領域3に分け(図11の(a)、(b)、(c))、それぞれ別々の Block RAM に格納し、3 つ並列に Template Matching 計算して高速化を計る(図 11 の(d),(e),(f))。



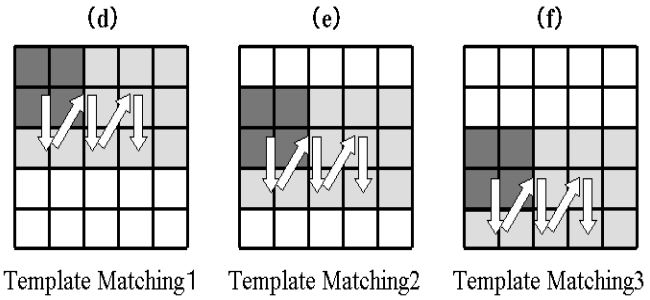


図 11 ： Template Matching の並列処理

続いて、テンプレートマッチングする上での処理の手順を図 12 と図 13 を用いて説明する。但し、図 12 のブロックのうち頭文字が B で始まっているものは Block RAM である。

ⅰ) 図 13-①の時にテンプレートデータを Local Memory から B_TG_RAMdw, B_TG_RAMup に書き込む。
ⅱ) 図 13-②の時に予測領域 1 のデータを Local Memory から B_IM1_RAMdw1,B_IM1_RAMup1,B_IM1_RAMdw2,B_IM1_RAMup2 に書き込む。
ⅲ) ⅱ)でテンプレートデータ以上のデータ量が書き込まれたら、テンプレートマッチング(Template Matching1)をしていく(図 13-③)。
ⅳ) 図 13-②が終わって(Block RAM=Max 状態)、図 13-④の時に予測領域 2 のデータを Local Memory から B_IM2_RAmdw1,B_IM2_RAMup1,B_IM2_RAMdw2,B_IM2_RAMup2 に書き込む。
ⅴ) ⅳ) でテンプレートデータ以上のデータ量が書き込まれたら、テンプレートマッチング(Template Matching2)をしていく(図 13-⑤)。

同じような要領で、予測領域 3 のデータの Block RAM への書き込みと Template Matching を行う。また、書き込んだデータ全てにおいて Template Matching 計算が終わったら、⑧、⑨、⑩でそれぞれ予測領域のデータを Block RAM に書き込む。
※ Block RAM を dw と up に分けた理由と、dw,up にそれぞれ 1,2 がある理由を説明する。

＜dw と up に分けた理由＞

Local Memory のデータ幅が 32 ビットであるのに対し、Block RAM 1 つのデータ幅は 16 ビ

ットである。故に、Local Memory から読み出されたデータの下位 16 ビットは dw に、上位 16 ビットは up に格納する。

＜dw,up にそれぞれ 1,2 がある理由＞

Block RAM 1 つのアドレス幅は 8 ビットであり、予測領域から取り出すデータはその値を越える。故に最初は、dw1,up1 にデータを格納し、全て埋まったら dw2,up2 に格納していく。

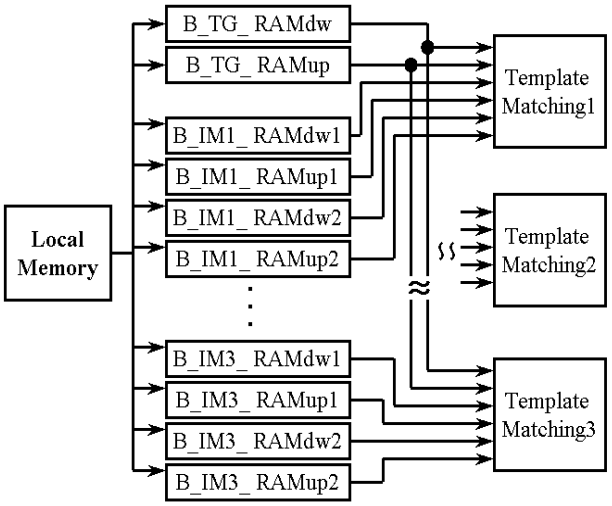


図 12：Template Matching における処理の流れ

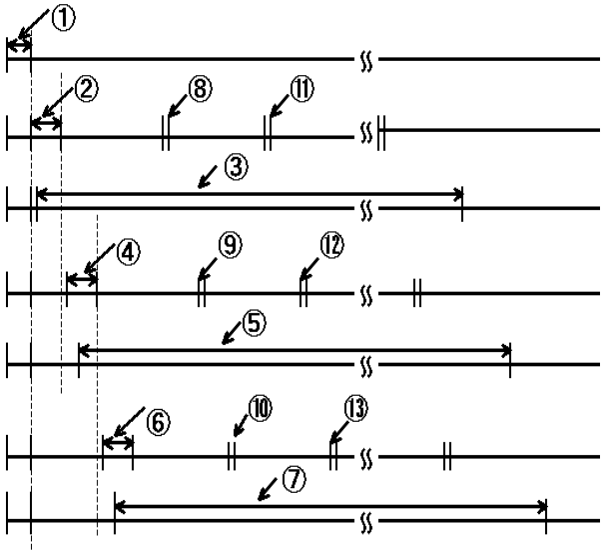


図 13：データ書き込みと Template Matching のタイミング図

5．実験

5-1．実験方法

今回、Level1 の入力圧縮画像(128×128)に対し、同 Level に圧縮したテンプレート(32×32)を、Tracking circuit によって予測領域 1 だけテンプレートマッチングし、それにかかる時間をシミュレーションから求めた。この時の設定は、top_height、bottom_height、 left_width、right_width がそれぞれ 1 で、addr_position、addr_in_position が 2　16(16進数)と 2 である。

5-2．回路規模と動作速度

Tracking circuit の論理合成を、ザイリンクス社製の論理合成ソフト FOUNDATION Ver.3.3i を用いて行った。表 1 には、FPGA「spartanⅡ2s200-FG456」を選択して論理合成を行った場合の回路規模・最大周波数・動作速度・処理に要するクロック数及び HDL 記述量を示した。

回路規模(gates)	126904
最大周波数(MHz)	36.498
処理に要するクロック数 (cycles/画面)	2440464
動作速度(ms)	73.9
HDL 記述量(lines)	1847

表 1 ： 回路規模・動作速度及び HDL 記述量

5-3．シミュレーション結果

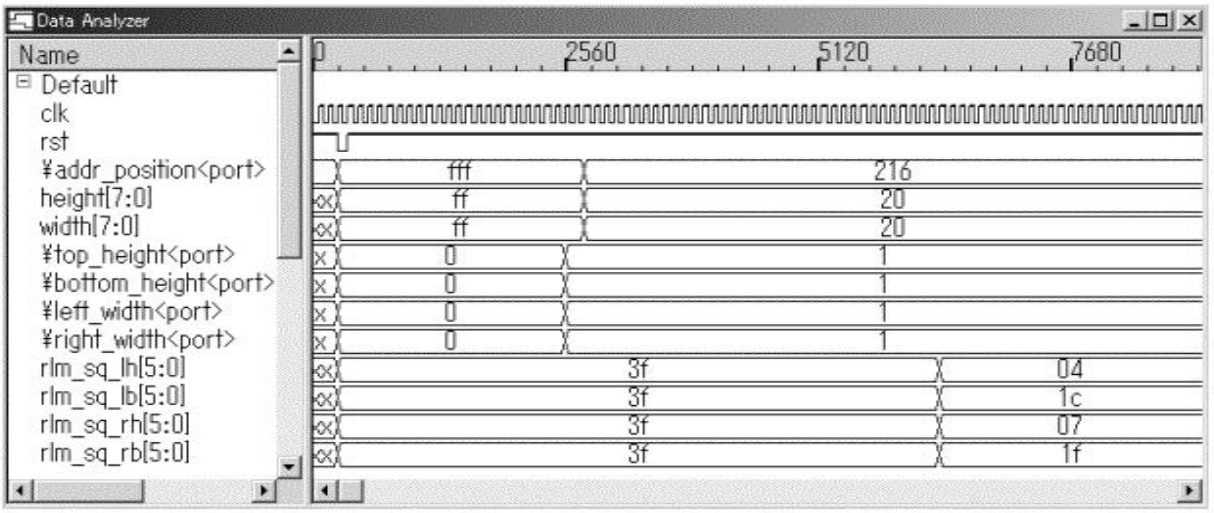


図 12：予測領域の決定

Tracking circuit のシミュレーションを論理合成ツールを用いて行った。図 12 は、addr_position をもとに予測領域を決定したものである。rlm_sq_lh、rlm_sq_lb、rlm_sq_rh、rlm_sq_rb は、予測領域の左上端、左下端、右上端、右下端の領域を示している。

6．まとめ

この実験では、圧縮 Level1 で予測領域 1 をテンプレートマッチングして 73.9(ms)かかった。この結果から Level2 と Level3 で予測領域 1 をテンプレートマッチングすると、それぞれ 19.8(ms)、4.6(ms)かかると予測される(Level1 にかかる時間の 1/4、1/16)。しかし、リアルタイムに追跡する為には、予測領域を 30(ms)以内でテンプレートマッチングする必要がある。これに対し、Level2 と Level3 はリアルタイム処理可能だが、Level1 は不可能である。Level1 で時間がかかったのはテンプレートマッチング計算に原因がある。そこで、テンプレートマッチングの並列処理を増やすことで高速化が考えられる。現在使用している HwModule に搭載されている FPGA は、14 個(2 個はテンプレート用、4×3 個は予測領域用に使用)の Block RAM を所持している。Block RAM を多く所持した FPGA を使用するか、Tracking circuit を複数の FPGA に実装し並列処理されることで、Level1 においてもリアルタイム処理が可能になる。

参考文献

[1] Mohammad Gharavi-Alkhansari,”A Fast Globally Optimal Al-gorithm for Template Maching Using Low-Resolution Pruning,”in IEEE TRANSACTIO NS ON IMAGE PROCESSING, Vol.10, No.4, Apri l, 2001.

[2] 高野茂, 新島耕一,“ Haarウェーブレットを用いた高速カラー画像検索システム, ”九州大学大学院システム情報科学研究科報告,Vol2 , No 2,pp.229-234,September 1997.

[3] 谷荻隆嗣,“ VLSIとディジタル信号処理, ”コロナ社, 1997

[4] 榊原進, “ウェーブレットビギナーズガイド,”東京電機大学出版局, 1995.

[5] 新島耕一,“ ウェーブレット画像解析, ”科学技術出版, 2000.