FPGA を用いた 3 玉モデルの実時間適合度計算

Real-time Evolutionary Optimization with Three-ball Model Using FPGA

○学　仁田　貴大（岡山大）　　羅　詩ケン (岡山大)
正　見浪　護（岡山大）　　正　松野　隆幸（岡山大）
正　戸田　雄一郎（岡山大）
Takahiro NITTA, Okayama University, piux45ja@s.okayama-u.ac.jp
Shiqian LUO, Okayama University
Mamoru MINAMI, Okayama University
Takayuki MATSUNO, Okayama University
Yuichiro TODA, Okayama University

Today, the introduction of robots is desired for exploration in unknown environments and investigations of disaster sites. Intelligent robot is required to work in an environment where it is constantly changing. In our laboratory, we have been studying visual servoing by stereo vision. Then, we are trying to introduce FPGA for shortening real-time measurement. FPGA stands for "Field Programmable Gate Array." It has already been applied in a wide range of fields such as communications, medical care, and analysis. FPGA can easily change integrated circuits and perform task processing in parallel. This can be expected to speed up the calculation of goodness of fit and increase the number of GA evolutions. However, it has not been fully introduced yet. In this report, we report the realization of the calculations of a three-ball model-based matching by FPGA.

Key Words: FPGA, Unrolling, Visual servoing

1　緒言

2020 年はコロナの流行により，リモートワークやオンライン授業，オンライン会議など社会のあり方が大きく変わっていきました．今までは人と人が直接会って交流をしてきました．しかし，今後は人と人とがネットを介して交流をしていく機会が増えていくと思います．また，人とロボットの交流も増えていくと思います．ロボットや AI は，より大量の情報，画像，動画など，さらなる処理能力の高速化が求められています．また，未知の環境での探索，災害現場の調査においてロボットの導入が望まれている．ロボットが常に変化する環境の中で作業するには，環境変化に適応できるリアルタイム制御のロボットが求められる．我々の研究室ではステレオビジョンによるビジュアルサーボイングの研究を行っている．我々はステレオカメラにより取得した視覚情報を用いた複眼立体認識によって，水中ロボットの制御を提案し，実現している．[1][2] 提案した実時間遺伝的認識手法は三つのカラー球からなる 3D マーカーを用意し，その位置・姿勢を遺伝的アルゴリズム（Genetic Algorithm. 以下 GA）によってオンラインで計算させ，適合度の最も高い遺伝子が持つ位置・姿勢情報を水中ロボットが認識した 3D マーカーの位置・姿勢として利用する方法である．他手法と異なる点は，(1)3D マーカーを複眼で認識している点 [3] と，(2) 動画像中の画像とモデルとの相関を GA の進化における適合度として用いることで対象物の位置・姿勢を実時間で計測している点 [4][5] である．さらに，動画像中の GA の収束性能に関する検証 [6] も行っている．現在は，CPU を用いて適合度の計算や GA を行っています．しかし，さらなる処理能力を実現するには不十分です．そこで我々は FPGA の導入を試みています．FPGA とは，Filed Programmable Gate Array の略です．FPGA は，デバイス内の論理ブロックを複数組み合わせて論理回路を実現する PLD（Programmable Logic Device）の一種であり，ユーザが手元でプログラムを書き換え可能という特徴があります．また，開発コストが安価で，消費電力が少ないなどの利点が多くあり，近年様々な分野で用いられてきています．現在使用している FPGA ポートは，図 1 に示す XILINX の Zynq UltraScale+ MPSoC ZCU104 です．

Fig.1 XILINX'Zynq UltraScale+ MPSoC ZCU104

2　FPGA の開発環境の構築

2.1　開発準備

FPGA を用いるためには，コンピュータの環境構築を事前にする必要がある．Ubuntu Linux 16.04 LTS(64bit) をインストールする．また，Vitis 統合ソフトウェアプラットフォームのインストールは，エッジ、クラウド，そしてハイブリッドの演算アプリケーションに対応する 1 つの統合されたプログラミング モデルが提供される．Vivado Design Suite は，C 言語でプログラムを設計するにあたって必要なツールとデザインの生産性が開発者側へと提供される．PetaLinux ツールは，ザイリンクスのプロセッシング システム上で組み込まれた Linux ソリューションをカスタマイズ，ビルド，およびデプロイするために必要なものをすべてが提供される．

2.2　OPEN CL と OPEN CV

2.2.1　OPEN CL

Open Cl(Open Computing Language) は、中央処理装置 (CPU), field programmable gate array (FPGA)，その他のプロセッサまたはハードウェアアクセラレータで構成される異種プラットフォーム間で実行されるプログラムを作成するためのフレームワークである．

2.2.2　OPEN CV と xf::open::cv

Open CV（Open Source Computer Vision Library）は，リアルタイムのコンピュータービジョンを目的としたプログラミング機能のライブラリである． 画像データは CPU から opencv によって読み取られ，Mat という名前のクラスに格納される． Mat

は，画像の情報と属性を保存する画像コンテナのようなものである． xf :: open cv は，Xilinx が提供する FPGA 開発を対象としている．画像情報を int 型として格納する cv :: mat とは異なり，r，g，b が読み取りやすくなっている． xf :: mat のデータ型は，クロックサイクルごとに処理するピクセル数と type パラメーターによって異なり，可能なデータ型は異なっている．

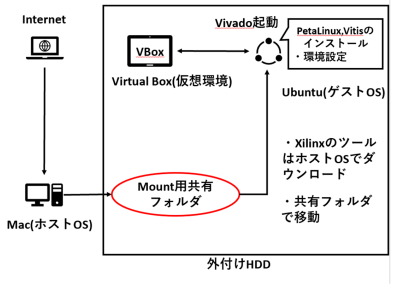

Fig.2 FPGA Development Environment

Table 1 xf::Mat Class - Available Data Types

Option	Number of bits per Pixel	Type
XF_32UC1	32	Unsigned
XF_32FC1	32	Float
XF_32SC1	32	Signed
XF_16CU1	16	Unsigned
XF_16SC1	16	Signed
XF_8UC4	8	Unsigned
XF_8UC3	8	Unsigned
XF_8UC3	8	Unsigned
XF_8UC1	8	Unsigned
XF_2UC1	2	Unsigned

$$wordwidth = 8 \times 3 \times 1 \quad (1)$$

最初の 8 ビットは B(Blue)，次の 8 ビットは G(Green)，最後の 8 ビットは R(Red) を表す．そして，以下のプログラムをピクセル用いることで，R，G，B のデータを読み取ることができる．

$$b = rgb\&0xff; \quad (2)$$
$$g = (rgb>>8)\&0xff; \quad (3)$$
$$r = (rgb>>16)\&0xff; \quad (4)$$

3 FPGA を用いた適合度計算

3.1 FPGA プログラムと高位合成

適合度の値を計算する方法について説明する．最初に CPU から画像データを取得し，位置と姿勢を与えて，外側の部分と内側の部分を含むモデルを計算する． 次に，モデルのサンプル点である r，g，b 値を読み取り，それを使用して明るさと色相を求める． 最後に，左右の画像の適応度関数を計算をする． また，位置と姿勢は，xsft = 22.66，ysft = 19.24，zsft = 339.99，n1 = 0.0052，n2 = 0.0354，n3 = 0.0234 に設定されている．
我々の目標は，左右のカメラから得られた画像をもとに FPGA 側でモデルを作成し，適合度計算を計算することである．しかし，プログラムを作成するにあたっていくつかの問題がある．その問題について次に説明する．

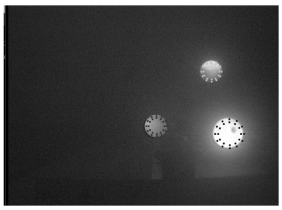

Fig.3 3-ball model in left image

3.2 プログラムの問題点と改善

FPGA を用いて画像を処理するにあたって一番の問題点は，計算ユニットの不足にある．左右の画像を処理して，計算するとなると計算ユニットが半分以上足りないという結果になる．そこで，まず我々が行ったことは左画像だけを使用するということである．単純に考えて半分の計算処理で済むと考えたからである．次に FPGA 側で全ての処理を任せるのではなく，CPU 側でも処理を行い，FPGA 側へその結果を送ることにした．また，画像の読み込み方法を stable 型から stream 型へと変更した．stable 型は，画像のデータを繰り返し読み取ることができるが，メモリの消費量が大きいという問題点があった．一方，stream 型は，変数が RAM として実装されているので画像のデータを一度しか読み取れないが，メモリの消費量は少なく大量のメモリを消費する適合度計算においては最適である．stream 型の画像の読み込み方について図 4 に示す．この結果，適合度の計算ができることを確認している．このときのフローチャートを図 5 に示す．

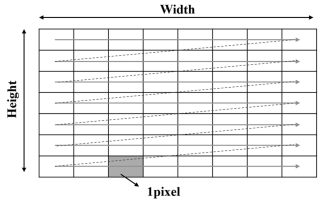

Fig.4 Image loading order in stream type

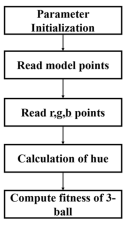

Fig.5 Flowchart of 3-ball fitness

4 動画像の導入

4.1 Gstreamer

まず Gstreamer をダウンロードする．Gstreamer とは，ストリーミングメディアアプリケーションを作成するためのフレームワークである．開発フレームワークでは、任意の種類のストリーミングマルチメディアアプリケーションを記述できるよう設計されている．Gstreamer フレームワークのベースとなる各種プラグインは、さまざまなコーデックおよびその他の機能を提供する．

今回は，カメラの初期化や設定，画像の読み込みを可能にするために V4L2 ツールを用いている..

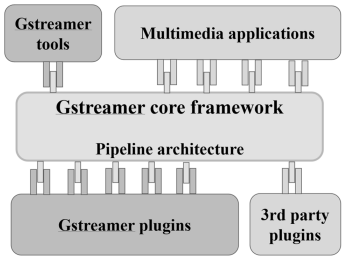

Fig.6 Overview of Gstreamer

4.2 複眼カメラを用いた動画像の出力

今回用いたカメラを図 7 に示す．カメラから読み込んだ画像を SD カードに保存するまでについて説明する．まず V4L2 ツールを用いてカメラ画像を読み込めるように設定し，CPU 側へと次々に画像を送り込んでいく．次に，DMA 転送により FPGA 側へと画像を送る．FPGA 側では，送られてきた画像に対して処理をおこなう．その後再び CPU 側へと DMA 転送する．そして，その画像を OpenCV を用いて，SD カードへと動画として保存する．この流れを図 8 に示す．ここに示す制御 PC は，FPGA への実行命令および出力画面として用いている．出力画面には FPGA で実行した結果が示される．基本的な流れは述べたとおりであるが，FPGA 側での処理に時間を費やしているという問題がある．この問題は最適化をすることにより解決しようと試みている．

Fig.7 USB camera(See3CAM_CU30_CHL_TC_BX) used

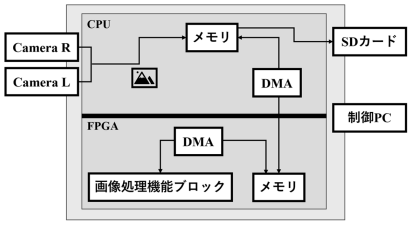

Fig.8 Camera image processing procedure

5 プログラムの最適化と並列処理

FPGA での処理を速くするために最適化について説明する．最適化の方法はいくつかあるが，我々は並列処理をすることにより最適化をしようと試みている．

5.1 ループ展開

ループ展開は，ループ内のプログラムコードを複数個記述することにより，ループ回数を減らすという手法である．図 9 に示すように係数を 2 でループ展開することにより，ループ本体の 2 個のコピーが作られ，それに伴いループ変数 (i+1) やループの繰り返しカウンタ (i+=2) が自動的に更新される．このとき，ループ内の演算量が増えているが，ループの回数を減らすことができている．ループを展開するには，ループの開始部分に#pragma HLS unroll [factor=N] を記述する必要がある．N は，ループを展開するときの係数である．

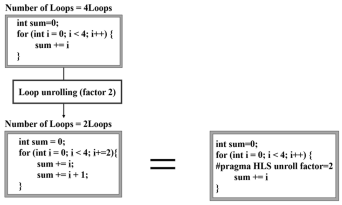

Fig.9 Loop unrolling processing

6 演算処理におけるループ展開の有効性の確認実験

6.1 実験内容

左右 1 枚の画像に対して，FPGA が遺伝子の数 (1,2,4,8,16,32,40) だけ適合度計算をおこなう．このとき，ループ展開により処理時間が短くなっているかについて検証する．それぞれループ展開なしの場合とループ展開ありの場合について，演算にかかった時間 [ms] を計測する．ループ展開は，全展開である．図 10 に今回用いた画像を示す．

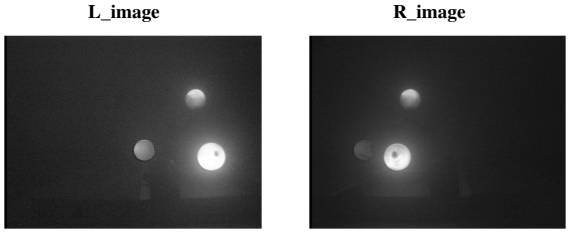

Fig.10 Left image and right image used in the experiment

6.2 実験結果

実験結果を表 2 に示す.

Table 2 Difference in arithmetic processing time in FPGA depending on the presence or absence of loop unrolling

	Time[ms]	
Number of Genes	Without Loop Unrolling	With Loop Unrolling
1	52.0	51.9
2	Failure	107.2
4	Failure	209.7
8	Failure	414.6
16	Failure	824.5
32	Failure	1644.3
40	Failure	2054.2

6.3 考察

ループ展開なしの場合だと遺伝子の数が 1 のみ成功し，他はすべてビルドができなかった．失敗した理由は FPGA 側でのメモリ容量の不足であった．成功した計測時間である 52[ms] は，適合度を 1 回計算するのにかかった時間であり，速いとはいえない．ループ展開ありの場合だとすべて成功し，演算時間を計算できた．遺伝子の数が 1 の場合は，ループ展開をしても値はほぼ変わらなかった．これは，遺伝子の数が 1 だと 1 回の演算で処理が終わるためであり，ループ展開なしの場合と同じ結果がでていることは正しいことだとわかる．遺伝子の数が 2 倍ずつ増えると計測時間も約 2 倍になっていることがわかる．この結果から，法則性が見られるので計測の仕方は問題ないといえる．ループ展開を用いることで同時に演算ができるようになり，メモリ容量の不足がなくなりビルドをすることができた．

参考文献

[1] 大西祥太，須浪唯介，西村健太，矢納陽，石山新太郎，見浪護，藤本勝樹 “MOS 制御知能を搭載した遠隔操作型水中ロボット (ROV) の自律制御化 (AUV) 技術 ”，第 57 回自動制御連合講演会 (2014)

[2] 矢納陽，大西祥太，米森健太，石山新太郎，藤本勝樹，見浪護，“ビジュアルサーボによる水中ロボットの位置・姿勢制御 ”，第 6 回コンピューテーショナル・インテリジェンス研究会 (2014)

[3] Song, W., Yu, F. Mae and Minami, M “3D visual servoing by feedforward evolutionary recognition, Journal of Advanced Mechanical Design,Systems, and Manufacturing, Vol.4, No.4(2010), pp.739-755.

[4] Suzuki,H., Minami,M., “Visual Servoing to catch fish Using Global/local GA Search”,

[5] Nishimura,K., Hou,S., Maeda,K., Minami,M, “Analyses on on-line evolutionary optimization performance for pose tracking while eye-vergence visual servoing”,Proceedings of 2013 IEEE international Conference on mechatronics and Automation(ICMA)(2013),pp.698-703.

[6] Yu,F., Minami,M., Song,W., “Eye-vergence visual servoing enhancing Lyapunov-stable trackability”,Artificial Life and Robotics, Vol.18,No.1-2(2013),pp.27-35.

[7] Nakamura,S., Yamada, D., Mukada,N., Myo,M., Minami,M.,“Development of Dual-eyes Docking System for AUV with Lighting 3D Marker ”, OCEANS 2018 MTS/IEEE Charleston, OCEANS 2018.

[8] 中村 翔, Yamada, D., Mukada,N., Myo,M., Minami,M.,“Development of Dual-eyes Docking System for AUV with Lighting 3D Marker ”, OCEANS 2018 MTS/IEEE Charleston, OCEANS 2018.

[9] XILINX, “ザイリンクス OpenCV ユーザーガイド ”, https://japan.xilinx.com/html_docs/xilinx2019_1/sdaccel_doc/fgu1544032152414.html, (2020-1-30)

[10] SDSoC Environment, “ハードウェア関数の最適化 ”, https://japan.xilinx.com/html_docs/xilinx2019_1/sdsoc_doc/optimizing-hardware-function-uww1504034429970.html, (2020-1-30)

[11] Infoscience Corporation, “GStreamer アプリケーション開発マニュアル 日本語訳 (0.10.25.1)” , http://oss.infoscience.co.jp/gstreamer/index.html, (2020-1-30)