FPGA ベース AmoebaSAT ソルバによる自動運転の複数タスクの高速化

井沼　佑亮†　　原　　祐子†

† 東京工業大学 工学院 情報通信系　〒152–8550 東京都目黒区大岡山 2 丁目 12–1
E-mail: †{inuma,hara}@cad.ict.e.titech.ac.jp

あらまし　近年，組込みシステムのアプリケーションは様々な処理から構成され，大規模・複雑化しており，各々の処理に対して専用回路を搭載することは回路規模の問題から困難である．本研究では， International Conference on Field Programmable Technology で開催されている自動運転コンテストの効率的な高速化手法を提案する．具体的には，そのコンテストで解決すべき課題である経路生成と物体検出を充足可能性問題（SAT）に帰着させ，FPGA ベースの SAT ソルバを設計し，前述の 2 つの処理に対する共通のアクセラレータとして応用する．これらの処理をハードウェア資源に乏しい小規模な FPGA で動作させるための効率的な問題定式化手法を示す．実験により，経路生成および物体検出のそれぞれにおいて，提案手法の有効性を確認した．

キーワード　充足可能性問題，自動運転，経路生成，物体検出，FPGA

1. 序　　論

近年，自動車やドローンなどの小型デバイスの自動走行技術が注目を集めている [1]．これらのアプリケーションにおいて，現在地点から目的地点までの最適経路を求める経路探索問題や，装着されているカメラから得られる画像から障害物を検出する物体検出問題は非常に基本的かつ重要なタスクである．各タスクに対する専用回路を利用することで効率良く処理できるが，対象のタスクのそれぞれに対して専用回路を利用しようとすると，総じて大きな回路面積を必要としてしまい，小型デバイスへの実装が困難になる．

本研究では，自動運転に関わる複数タスクをそれぞれ充足可能性問題 (Satisfiability Problem; SAT) に帰着し，FPGA 実装した SAT ソルバ [2] によって解くことによって，複数タスクに対する “共通のアクセラレータ” として実現する手法を提案する．本論文では International Conference on Field Programmable Technology (ICFPT) の FPGA デザインコンペティション および相磯秀夫杯 FPGA デザインコンテストを対象にする．このコンテストではミニチュア車両を FPGA で自動制御し，定められた順路のコース上に存在する障害物を避けながら制限時間以内に走行するものである．そのアプリケーションを構成している経路生成および物体検出を SAT 問題 に帰着し，ハードウェア実装された AmoebaSAT でそれぞれの問題を高速に解く．AmoebaSAT の実装には高位合成を用い，ループのパイプライン化や配列分割の最適化を行うことで，さらにハードウェア化の有効性を活用できる．実験では各タスクの処理速度や得られた解の質について評価し，本研究の有効性を確認した．

2. AmoebaSAT を用いた経路生成手法

本研究では，自動運転に関する処理をそれぞれ充足可能性問題（Satisfiability Problem; SAT）に帰着させ，それらを高速なハードウェア SAT ソルバを用いて解くことにより，効率的に複数処理を高速化を図る．SAT ソルバには Aono らによって提案された AmoebaSAT [2] を高位合成技術を用いてハードウェア実装し，その他の処理は CPU 上で動作するソフトウェアとして実装する．本論文では，経路生成と物体検出の 2 つの課題を SAT にそれぞれ応用する．本章では，経路生成を SAT の派生である Satisfiability Modulo Convex (SMC) に，次章では物体検出を SAT に帰着する方法をそれぞれ説明する．

2.1 Satisfiability Modulo Convex (SMC)

経路生成問題とは，現在地点から目的地点まで，障害物と接触しないような走行路（パス）を求める問題である．経路生成問題を AmoebaSAT で解くためには，問題を SAT の論理式で表現する必要がある．そのために， Shoukry らにより考案された Satisfiability Modulo Convex (SMC) 式を導入する [4]．SMC 式は Boolean 変数，論理和，論理積，凸条件式などが混合したものであり，形式的には以下のように表される．

$$\begin{aligned} formula &:= clause\{\wedge clause\}* \\ clause &:= literal\{\vee literal\}* \\ literal &:= bool\_var \mid \neg bool\_var \mid 1 \mid 0 \mid conv\_cn \\ conv\_cn &:= equality \mid inequality \\ equality &:= affine\_function(x) = 0 \\ inequality &:= convex\_function(x)\ relation\ 0 \\ relation &:= < \mid \leq \end{aligned} \tag{1}$$

ここに， $*$ は 0 回以上の繰り返し， $bool\_var$ は Boolean 変数，$affine\_function$ はアフィン関数， $convex\_function$ は凸関数を表す．SMC 式の解き方を，次式 φ を例に説明する．

$$\begin{aligned} \varphi =& (b_1 \vee b_2) \wedge (\neg b_1 \vee 3x_1 + 4x_2 + 2 < 0) \\ & \wedge (b_2 \vee x_1^2 + 4x_2^2 - 4 < 0) \\ & \wedge (\neg b_2 \vee \neg b_3 \vee 2x_1^2 + 5x_2 - 12 < 0) \end{aligned} \tag{2}$$

まず，φ に含まれる凸条件式（式 (1) の $conv\_cn$ に対応する

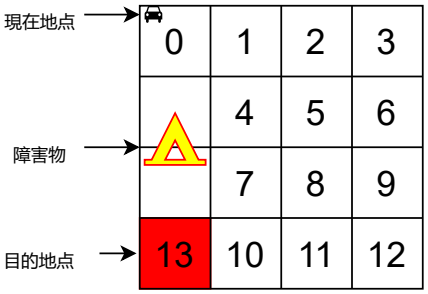

図 1　グリッド分割の例（[7] より引用）.

部分）を新たな Boolean 変数 a_1, a_2, a_3 で置換し，φ_B とする．

$$\varphi_B = (b_1 \vee b_2) \wedge (\neg b_1 \vee a_1) \wedge (b_2 \vee a_2) \wedge (\neg b_2 \vee \neg b_3 \vee a_3) \quad (3)$$

ここで，φ_B はリテラルが論理和と論理積で結ばれた式であるため，$\varphi_B = 1$ を満たす変数割当て（解）を見つける問題は SAT であり，SAT ソルバである AmoebaSAT で解くことができる．解は複数存在しうるが，ここでは次の割当てが得られたと仮定する．

$$b_2 = b_3 = a_3 = 0,\ b_1 = a_1 = a_2 = 1 \quad (4)$$

次に，SMC 式を SAT 式に変換する際に新たに追加した変数のうち，1 が割当てられた変数に対応する条件式を列挙する．その後，それらの条件式を満たす x_1, x_2 を求める．例えば，(4) では a_1 と a_2 に 1 が割当てられており，それぞれに対応する条件式は $3x_1 + 4x_2 + 2 < 0$ と $x_1^2 + 4x_2^2 - 4 < 0$ である．これらを満たす x_1, x_2 がなければ，解 (4) が誤りであるため，もう一度 φ_B を解き直す必要がある．その際，解で 1 が割当てられた全ての変数に否定をつけ論理和で結んだ節（この例の場合は $(\neg b_1 \vee \neg a_1 \vee \neg a_2)$ ）を追加することにより，同一の解を回避することができる．

2.2　経路生成問題の SMC 式による表現

続いて，経路探索問題を SMC 式で表現する方法を説明する．まず，探索する領域を図 1 のようにグリッドに分割する．この例は，左中央にある障害物を避けながら，地点 0 から地点 13 への経路を求める問題を表している．ここで，グリッドの数が多いほどより良いパスが得られるが，その分 SMC 式およびそれを変換した SAT 式が複雑化する．次に，以下の 2 種類の変数 b, x を定義する．

- $b_t^i \in \{0, 1\}$: 時刻 t に車体が領域 i に存在するならば，かつそのときに限り真となる Boolean 変数.
- $x_{2t}, x_{2t+1} \in \mathbb{R}$: 時刻 t に車体が存在する x, y 座標.

そして，これらの変数を含む SMC 式 φ を以下で定義する．

$$\varphi = b_0^0 \wedge b_L^{M-1} \quad (5)$$

$$\wedge \bigwedge_{\substack{k\in\{0,\dots,L-1\},\\ i\in\{0,\dots,M-1\}}} \left(\bigvee_{i'\in\Pi(i)} b_k \to b_{k+1}^{i'} \right) \quad (6)$$

$$\wedge \bigwedge_{k=0}^{L} \left(\sum_{i=0}^{M-1} b_k^i = 1 \right) \quad (7)$$

$$\wedge (x_0 = x_{init} \wedge x_1 = y_{init}) \quad (8)$$

$$\wedge \bigwedge_{\substack{k\in\{0,\dots,L\},\\ i\in\{0,\dots,M-1\}}} \left(b_k^i \to (x_{2k}, x_{2k+1}) \in R_k \right) \quad (9)$$

ここに，L は目的地点に到達する時刻，M はグリッドの総数，$x_{init}, y_{init} \in \mathbb{R}$ はそれぞれ初期 x, y 座標，$R_k \subset \mathbb{R}^2$ は領域 k の点全体からなる集合である．また，$\Pi(i)$ は領域 i と領域 i に隣接する領域のうち，目的地点から上下方向に遠ざかるものを除いた領域番号の集合である．例えば，図 1 の分割では，$\Pi(1) = \{1, 2, 4\}, \Pi(7) = \{7, 8, 10\}$ などとなる．φ の各項は次の制約を表現している．

- 項 (5) は時刻 0 においては車体は領域 0 に存在して，最後の時刻においては目的の領域に存在することを表す.
- 項 (6) は車体は 1 単位時間では隣り合った領域にのみ移動できる制約を表す.
- 項 (7) は車体が存在する領域は 1 つである制約を表す.
- 項 (8) は初期座標を表す.
- 項 (9) は領域と実際の座標を関連付けるものである.

節 2.1 で紹介した手法を用いて SMC 式 φ を解き，得られた x を順に結ぶことで，経路を得ることができる．

3.　AmoebaSAT を用いた物体検出手法

3.1　物体検出問題の SAT 問題としての表現

物体検出問題を SAT に帰着するため，特徴量マッチングという手法を用いる．これは，2 つの画像から検出された特徴点を適切に対応させ，物体を検出する手法である．以下では，検出したい物体のみが写った画像をテンプレート画像，車体に装着されているカメラから得られる画像を入力画像と定義する．まず，この 2 つの画像双方に対して特徴量検出を CPU 上で行う．特徴量検出アルゴリズムは多数提案されているが，今回は計算コストが軽量な ORB [5] を採用した．テンプレート画像，入力画像から抽出された特徴点の集合をそれぞれ $F_T = \{p_1, \dots, p_N\}, F_C = \{q_1, \dots, q_M\}$ とし，次の 4 種の変数を定義する．

- $b_{ij} \in \{0, 1\}$: テンプレート画像の特徴点 p_i が入力画像の特徴点 q_j と対応するならば，かつそのときに限り真となる Boolean 変数.
- $f_p \in \mathbb{R}^k$: 特徴点 p の特徴ベクトル.
- $(x_i, y_i) \in \mathbb{R}^2$: テンプレート画像の特徴点 p_i の座標.
- $(u_j, v_j) \in \mathbb{R}^2$: 入力画像の特徴点 q_j の座標.

これらの変数を用いて，次の論理式 φ を定義する．

$$\varphi = \bigwedge_{1\le i\le N} \bigvee_{1\le j\le M} \left(b_{ij} \to \bigwedge_{1\le j'\le M, j'\ne j} \neg b_{ij'} \right) \quad (10)$$

$$\wedge \bigwedge_{1\le j\le M} \bigvee_{1\le i\le N} \left(b_{ij} \to \bigwedge_{1\le i'\le N, i'\ne i} \neg b_{i'j} \right) \quad (11)$$

$$\wedge \bigwedge_{1\le i\le N} \sum_{1\le j\le M} b_{ij} = 1 \quad (12)$$

ここで，φ の各項は，特徴点の対応付けにおい次のて基本的な

条件を表している．

- 項 (10) は，テンプレート画像の特徴点が対応する入力画像の特徴点は唯一つである．
- 項 (11) は，入力画像の特徴点が対応するテンプレート画像の特徴点は唯一つである．
- 項 (12) は，テンプレート画像の特徴点は全て入力画像のいずれかの特徴点に対応する．

上記の論理式 φ は特徴点の位置関係やその特徴量を考慮していない．したがって，特徴点の位置関係の制約を追加するために，次の 2 つの集合 V_T, H_T を定義する．

$$V_T = \{(p_i, p_{i'}) \in F_T \times F_T \mid |x_i - x_{i'}| < \varepsilon_V, y_i < y_{i'}\}, \quad (13)$$

$$H_T = \{(p_i, p_{i'}) \in F_T \times F_T \mid |y_i - y_{i'}| < \varepsilon_H, x_i < x_{i'}\}. \quad (14)$$

ここで $\varepsilon_V, \varepsilon_H$ は閾値であり，実験により適当に定める．式 (13) は，テンプレート画像の 2 つの特徴点 $p_i, p_{i'}$ が，相対的に上下にこの順に並んでいる場合に $(p_i, p_{i'}) \in V_T$ となることを表している．同様に式 (14) は相対的に左右にこの順に並んでいる場合に $(p_i, p_{i'}) \in V_H$ となる．これらを用いて，各 $(p_i, p_{i'}) \in V_T$ と $j \in \{1, \ldots, M\}$ に対して $b_{ij} \to \bigvee_{q_{j'} \in V_{T(j)}} b_{i'j'}$ という節，および，各 $(p_i, p'_i) \in H_T$ と $j \in \{1, \ldots, M\}$ に対して $b_{ij} \to \bigvee_{q_{j'} \in V_{H(j)}} b_{i'j'}$ という節を先述の論理式 φ に追加する．ここで，$V_{T(j)}, V_{H(j)}$ は，V_T, V_H と同様の定義を入力画像に適用したものであり，それぞれ入力画像において q_j の下，右にある特徴点からなる集合である．これにより，テンプレート画像中で上下・左右に並んでいる特徴点が対応付く入力画像中の点もまた，上下・左右に並んでいることを表す．

3.2 論理式サイズの縮小手法

上記の手法によって作られた論理式は非常に肥大化する可能性がある．論理式の肥大化は，求解にかかる時間の増大を招く．さらにハードウェア SAT ソルバは予め解くことのできる論理式サイズの上限が FPGA のサイズによって決まるため，巨大な論理式は入力することさえも不可能である．そこで本節では，論理式のサイズを縮小するための 3 つの手法を説明する．

特徴量の考慮: 特徴点の特徴量を考慮するためには，論理式に適当な節を導入する方法もあるが，閾値 ε を適当に定め，不等式

$$|f_{p_i} - f_{q_j}| \geq \varepsilon \quad (15)$$

を満たす特徴点 p_i, q_j は対応しないという事実を用いて，予め $b_{ij} = 0$ と決定する．これにより，論理式から変数 b_{ij} を除去でき，さらに $\neg b_{ij}$ を含む節は必ず真となることから，そのような節も除去することができる．条件の多くは $b_{ij} \to \cdots$ という形をしていることから，この手法によって多くの節を除去できることが期待される．

近接する特徴点の一体化: 前節で定義した物体検出問題を表現する論理式は，画像に含まれる特徴点の個数によって変数や節の個数が増減する． ORB 特徴点検出 [5] を行ったところ，特徴点が局所的に集中する場合が多く見られたため，本研究では十分近くにある特徴点同士を 1 つの点にまとめ，論理式のサイズの縮小を試みた．まず，特徴点 p_i をランダムに選び，その近くに存在する特徴点からなる集合を $P_{p_i} = \{p_{i_1}, \ldots, p_{i_n}\}$ とする．この P_{p_i} は座標が (x_i, y_i) であり，特徴量は $F_i = \{f_{p_i}, f_{p_{i_1}}, \ldots, f_{p_{i_n}}\}$ であるような 1 つの “特徴点” とみなす．このとき，p_i と P_{p_i} に含まれる特徴点を，まとめられた特徴点と呼ぶこととする．まとめられることのできる特徴点が画像中に存在しなくなるまで上記の操作を繰り返す．このようにして新たに構築した “特徴点” に対しても同様に前節で述べた論理式 φ を考えることができる．ただし，式 (15) は，任意の $p \in P_{p_i}, q \in P_{q_j}$ に対して $|f_p - f_q| \geq \varepsilon$ となる場合にのみ $b_{ij} = 0$ と決定するように修正する．

色情報の利用: 更に論理式のサイズを縮小するため，色情報を用いる．例えば，コンテストで検出が必要な物体の 1 つには矢印があるが，この物体は赤を基調としている．そのため，入力画像の特徴点で，周囲に赤色がない点はテンプレート画像の特徴点と対応することはないと考えられる．これにより，入力画像の特徴点の個数を減らすことができ，結果的に論理式に含まれる変数や節の個数が減少する．

3.3 複数の種類の物体への対応

このようにして作った論理式 φ によって，入力画像に対して 1 種類の障害物を検出することができるが，コンテストでは複数の種類の物体を検出する必要がある．複数種類の物体を検出するためには，各物体に対して論理式を構築し，それらを順に解いていくという方法が考えられる．しかし，各論理式のサイズが小さい場合には，式 (16) に示す通り一纏めにすることで一度に複数種類の物体検出問題を解くことができる．

検出したい物体 $1, \ldots, N$ に対して，先述の方法によって作られた論理式を $\varphi_1, \ldots, \varphi_N$ とする．$a_1, \ldots, a_N$ を Boolean 変数とし，次の論理式 φ_{all} を考える．

$$\varphi_{all} = (a_1 \vee \cdots \vee a_N) \wedge (\neg a_1 \vee \varphi_1) \wedge \ldots (\neg a_N \vee \varphi_N). \quad (16)$$

この式においては，物体 i が検出されたとき，かつそのときに限り a_i が真になる．従って，φ_{all} を AmoebaSAT で解いた後，$a_1, \ldots, a_N$ の真偽値の割当てを確認することにより，SAT 問題を 1 度解くだけで複数種類の物体の検出が可能になる．

4. 実装と評価

本章では，前述のアルゴリズムを FPGA 実装した詳細について述べる．実装には，FPGA カーキット ad-refkit [3] を用いた．このキットでは，FPGA ボード Xilinx Zybo Z7-20 が採用されている．このボードには Zynq-7020 という SoC FPGA が搭載されており，CPU 上では Ubuntu 18.04 LTS が動作する．本章では，この FPGA に実装したアーキテクチャの全体図を説明し，最後に評価結果を示す．

4.1 システムの概要とアーキテクチャ

図 2 に，自動運転システムの全容のうち，本論文で実現したシステム概要を示す．AmoebaSAT のハードウェア設計は高位合成ツール Vivado HLS 2018.3 を用いて C 言語ベース設計を行い，以降の設計は Vivado 2018.3 を用いた．AmoebaSAT の設計時には PIPELINE プラグマによるパイプライン化と ARRAY

表 1　各リソースの使用量と使用割合

リソース名	BRAM_18K	DSP48E	FF	LUT
使用量	134	0	5,310	36,484
割合	47%	0%	4%	68%

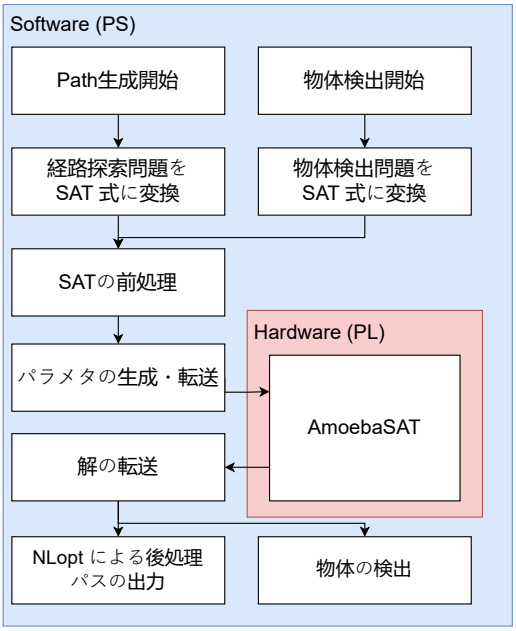

図 2　実装したアーキテクチャの全体図

PARTITION プラグマによる配列分割を行い最適化を行った．経路生成や物体検出が開始されると，第 2 章や第 3 章で説明した手法を用いて論理式を生成する．生成された論理式は，前処理を行うことで，更にサイズが削減できることがわかっている [7]．また，今回使用する AmoebaSAT は 3-SAT のみを扱うため，その変換も事前に行う必要がある．前処理した論理式に対して， AmoebaSAT の動作に必要なパラメタを生成する．ここまでの過程は全てソフトウェアとして CPU 上で実行される．AmoebaSAT はハードウェア実装されているため，これらのパラメタや SAT 問題の解を転送するために，AXI4-Lite インタフェースを使用した．AmoebaSAT を用いて解を求めた後，経路生成については 2.1 節で説明した通り，対応する変数が 1 となっている条件式を満たす x を求める必要がある．本研究では，無償公開されている最適化ライブラリ NLopt [6] を使用し，CPU 上でソフトウェアとして後処理する．物体検出では，物体が存在する物理的な座標を算出し，それが車体の正面にあるのか（つまり避けるべき障害物であるのか）を判定する．例えば，右車線にある物体は避ける必要がないので，障害物に該当しない．AmoebaSAT のハードウェア実装にかかる各リソースの使用量と使用割合は表 1 の通りとなった．

4.2　評　　価

まず提案した経路生成アルゴリズムの評価のため，既存手法である Informed-RRT* [8] との比較結果を表 2 に示す．ここで，表中の 3 つの指標は次のように定義される．両アルゴリズムは得られた中間点を結んだ線分によって経路を表現する．これらの線分の長さの総和を「長さ」，同じ中間点を共有する 2 線分の成す角度の平均を「滑らかさ」，総経路の中で最も障害物に接近したときの障害物との距離を「安全性」と定義する [9]．ただし，「長さ」「安全性」の単位は探索する領域の 1 辺を 1 としたときの距離である． Informed-RRT* は実行時間を大きくするほどより短い経路を出力するアルゴリズムであるため，提案手法の実行時間と同じ時間だけ実行させた場合に得られた経路を比較対象とした．この条件の下，図 1 の例において両アルゴリズムを 10 回試行し，その平均値を評価した．提案手法が生成した経路は Informed-RRT*のそれよりは長くなったが，同程度の滑らかさで，より安全な経路を探索できていることが分かる．提案手法には，各線分の長さや線分同士の角度を制限する項を追加することで，更に良いパスを得られる可能性がある．

表 2　Informed-RRT*との比較（「長さ」は小さい値が，その他は大きい値が良い結果を表す）

	長さ	滑らかさ	安全性
Informed-RRT*	1.514	2.350	0.1646
提案手法	1.784	2.383	0.2100

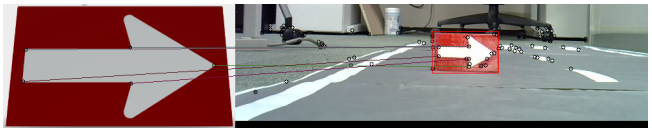

図 3　検出された物体の例

次に，提案した物体認識アルゴリズムの評価のため，各物体が写った画像を合計 1,994 枚用意した．これは，車体に装着された Web カメラを用いて実際に物体が写った写真を何枚か撮影した後，輝度や明るさの調整やノイズの追加などを行い生成した．これらの画像に対して物体検出アルゴリズムを動作させたところ，秒間 4.42 枚の処理が可能であった．以前のコンテストで同様の処理時間で好成績を収めているチームも存在しており，十分な処理速度であると考えられる [10] [11]．検出例を図 3 に示す．

5.　結　　論

本研究では，ハードウェア SAT ソルバである AmoebaSAT を経路生成や物体検出の 2 つの処理の高速化に応用する手法を提案した．これらの問題をそれぞれ SAT に帰着させることで，FPGA 実装した AmoebaSAT で高速に解くことができる．実験により，経路探索問題においては，既存のアルゴリズムと同等の経路を得ることができることを確認した．さらに物体検出問題においては，ミニチュアの FPGA カーが走行時に障害物を回避するためには十分な速度での検出が可能であることを確認した．

今後の課題として検出精度を上げることが挙げられる．特に歩行者を模した人形の検出の際には，他の物体と異なり際立った色が含まれていないため，第 3.2 節で述べた色情報の利用による論理式サイズの縮小が困難である．

謝　　辞

本研究の一部は，科学研究費補助金（挑戦的萌芽研究 JP20K21789 および基盤研究 (B) JP20H04154）の支援による．

文　　献

[1] D. Kyriazis and T. Varvarigou, “Smart, Autonomous and Reliable Internet of Things,” *Procedia Computer Science*, vol. 21, pp.442-448, 2013.

[2] A. H. N. Nguyen, M. Aono and Y. Hara-Azumi, “FPGA-Based Hardware/Software Co-Design of a Bio-Inspired SAT Solver,” *IEEE Access*, vol. 8, pp. 49053-49065, 2020.

[3] Y. Kudo, A. Takada, Y. Ishida and T. Izumi, “An SoC-FPGA-Based Micro UGV with Localization and Motion Planning,” *Proc. of Int’l Conf. on Field-Programmable Technology*, pp.469–472, 2019.

[4] Y. Shoukry, P. Nuzzo, A. L. Sangiovanni-Vincentelli, S. A. Seshia, G. J. Pappas and P. Tabuada, “SMC: Satisfiability Modulo Convex Programming,” *Proc. of the IEEE*, vol.106, no.9, pp.1655–1679, 2018.

[5] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, “ORB: An Efficient Alternative to SIFT or SURF,” *Proc. of Int’l Conf. on Computer Vision*, pp.2564–2571, 2011.

[6] Steven G. Johnson, “Stevengj/Nlopt: Library for Nonlinear Optimization, Wrapping Many Algorithms for Global and Local, Constrained or Unconstrained, Optimization,” *GitHub*, https://github.com/stevengj/nlopt, (2022 年 8 月 24 日参照)

[7] 井沼佑亮, 原祐子, “AmoebaSAT を用いた効率的な自動運転アクセラレータ,” 電子情報通信学会技術研究報告 *IEICE technical report:* 信学技報, vol. 121, no. 412, VLD2021-90, pp. 75-80, 2022 年.

[8] J. D. Gammell, S. S. Srinivasa and T. D. Barfoot, “Informed RRT*: Optimal Sampling-based Path Planning Focused via Direct Sampling of an Admissible Ellipsoidal Heuristic,” *Proc. of Int’l Conf. on Intelligent Robots and Systems*, pp.2997–3004, 2014.

[9] S. Hosseininejad and C. Dadkhah, “Mobile Robot Path Planning in Dynamic Environment Based on Cuckoo Optimization Algorithm,” *Int’l Journal of Advanced Robotic Systems*, vol. 16, no. 2, 2019.

[10] R. Yamamoto et al., "Development of Autonomous Driving System based on Image Recognition using Programmable SoCs," 2021 International Conference on Field-Programmable Technology (ICFPT), 2021, pp. 1-4.

[11] A. Kojima, "Autonomous Driving System implemented on Robot Car using SoC FPGA," 2021 International Conference on Field-Programmable Technology (ICFPT), 2021, pp. 1-4.