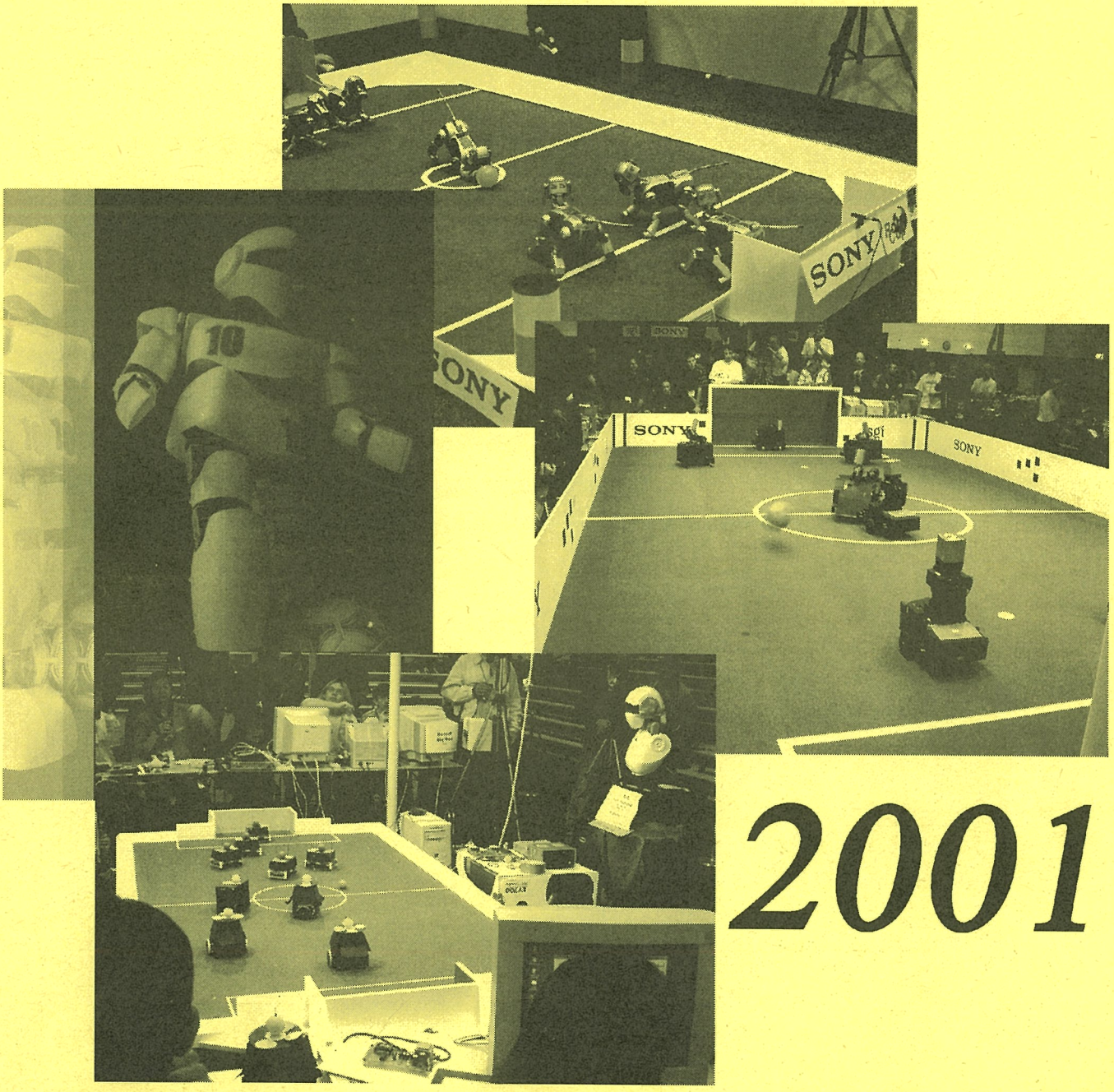
人工知能学会
第12回 SIG-Challenge研究会



2001

RoboCup

2001年4月27日
福岡工業大学
（第4回ロボカップジャパンオープン2001）

目次

方策勾配法を用いた移動ロボットの行動計画法
－問題の多様性への対応－

Motion Planning of Mobile Robots by a Policy Gradient Method
- Application to Various Problems-

五十嵐治一
Harukazu Igarashi

近畿大学工学部（広島県東広島市）
School of Engineering, Kinki University
igarashi@info.hiro.kindai.ac.jp

Abstract

In a previous paper, we proposed a solution to motion planning of a mobile robot. That approach is a reinforcement learning approach based on a policy gradient method. In this paper, we consider possibility whether this approach can be applied to various motion planning problems of mobile robots under the environments of multiple robots, moving obstacles, and partially observed Markov decision processes.

1 はじめに

自律移動型ロボットの軌道・経路計画問題に関しては，様々な方法が提案されてきた[1]－[10]．しかし，現実世界における軌道・経路計画問題は多目的多制約の計画問題となるのが一般的である．

その中で，従来，移動型ロボットの経路計画においては，ポテンシャル法，スケルトン法，空間分割法といった手法[4,7,10]が提案されてきた．これらの手法は主として経路の最短性と障害物回避との観点に重点を置いており，対象とする問題の前提条件やタスクの追加・変更に対しては，アルゴリズムやプログラムの大幅な修正を余儀なくされる場合もしばしば見受けられた．

そこで，著者は，ロボットの軌道・経路計画問題の分野において，柔軟で一般性があり，かつ実現が容易な手法を確立することを目的として，軌道・経路計画問題を各時刻における離散最適化問題に帰着させる方式を提案してきた[11]．さらに，その最適化問題で用いた目的関数中の重み係数の値を，強化学習の一種である方策勾配法(policy gradient method)により自動学習させるという手法を提案した[12][13]．

ただし，これまでの応用例では，単数ロボットが静止障害物の間を移動するという単純な場合のみを考えてきた．かつ，障害物の形状や配置によっては，局所的にトラップされてしまう場合も観測された．

そこで，本報告では，複数台ロボットの軌道・経路計画を立案する場合や，障害物が移動する場合，迷路のように局所的にトラップされやすい環境の場合，行動決定が部分観測マルコフ決定過程となる場合への対応を考察する．なお，本研究では，計画立案者の存在と，軌道・経路に対する報酬を与えるユーザ，または，ユーザの報酬と等価な価値基準が存在することを前提としている．

2 行動計画の基本アルゴリズム

2.1 目的関数

前章でも述べたように，自律移動型ロボットの軌道・経路計画問題（以下では，両者を合わせて，”行動計画問題”と称する）には，たとえ，既知環境下であっても，制約条件，前提条件，ユーザの要求する仕様[5][11]等による多様性が存在する．その多様性が，走行誘導のアルゴリズムを複雑にしている要因と考えられる．本章では，すでに著者が提案している行動計画方式[12][13]を簡単に説明する．

まず，次の目的関数 $E_v(v_t; r_t, v_{t-1}, r_{goal})$ を考える：

$$E_v(v_t; r_t, v_{t-1}, r_{goal}) = b_1 E_{goal} + b_2 E_{smth} + b_3 E_{clsn} \quad (1)$$

ここで r_t は離散時刻 t におけるロボットの位置，v_t は時刻 t におけるロボットの速度ベクトル，r_{goal} はゴール地点の位置である．この目的関数は，時刻 t において速度ベクトル v_t の値を選択した場合の不適当な度合い（ペナルティの量）を表現している．

最初の項 $E_{goal}(v_t, r_t, r_{goal})$ はロボットがゴール地点の方向へ進むことを要求している引力項である．具体的には

次の関数を用いた.

$$E_{goal}(v_t; r_t, r_{goal}) \equiv \mathrm{sgn}[G(v_t)] \cdot G(v_t)^2 \quad (2)$$

ただし，関数 $G(v_t)$ は次の式で定義されている.

$$G(v_t) = \left\| r_{goal} - r'_{t+1}(v_t; r_t) \right\| - \left\| r_{goal} - r_t \right\| \quad (3)$$

上式中，時刻 $t+1$ の予測位置 r'_{t+1} は，時刻 t におけるロボットの位置 r_t，時刻 t におけるロボットの速度 v_t，時間間隔 Δt により，以下のように定義されている.

$$r'_{t+1} \equiv r_t + v_t \Delta t \quad (4)$$

式(1)の第2の項 $E_{smth}(v_t, v_{t-1})$ は滑らか項であり，次式で定義する.

$$E_{smth}(v_t; v_{t-1}) = \left\| v_t - v_{t-1} \right\|^2 \quad (5)$$

ここで $\| v \|$ はベクトル v のユークリッドノルムを表す. この項は速度変化が少ない滑らかな軌跡が望ましいことを表している.

式(1)の最後の項 E_{clsn} は障害物との衝突を避ける斥力項であり，次式で定義する.

$$E_{clsn}(v_t; r_t) = \begin{cases} D_{clsn} & \text{if } Dist(r'_{t+1}) < 0 \\ -Dist(r'_{t+1})^2 & \text{if } 0 < Dist(r'_{t+1}) < R \\ -R^2 & \text{if } Dist(r'_{t+1}) > R, \end{cases} \quad (6)$$

ここで $Dist(r)$は，位置 r おいて障害物や壁までの最短距離を表した関数である. 本手法では障害物や壁の形状と位置とを正確に記述した環境地図を使用することを前提にしている. ただし，負の値は障害物等の内部であることを表しており，ロボットが障害物や壁と衝突しないように D_{clsn} の値は，十分大きな値に設定する. 式(6)の斥力項はロボットが障害物や壁からの距離が R 以内であれば，距離の2乗に比例した斥力が働くことを意味している.

2.2 探索空間

本来，目的関数 E_v は速度ベクトル v_t の連続関数であるが，ここで，速度ベクトルの探索空間を離散化し，原点を中心とする円内に探索範囲を限定する. 速度ベクトルの空間を離散化するのは，探索空間の分割数により，探索精度と処理速度とを制御することができるからである. すなわち，探索精度の方を優先したい場合には分割を細かくし，逆に，処理速度の方を優先したい場合には分割を粗くすれば良い. 探索空間の離散化の例を図1に示す.

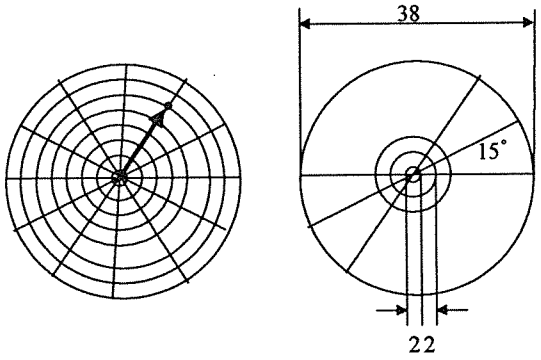


図1 速度ベクトルの探索空間の一例

2.3 期待報酬値と確率的政策

E_v を最小化して得られる軌道・経路は，式(1)中の重み係数$\{b_k\}(k=1,2,3)$に大きく依存する[11]. そこで最適な値を決定するために強化学習を用いた方式を考案した[12]. 本節と次節とで簡単に述べる.

まず，学習の目的を期待報酬の最大化と定める. 移動ロボットの行動計画の場合，軌道 u に与えられる報酬 $R(u)$の期待値を最大化するように，各時刻 t における速度ベクトル v_t を選択する方策 π の中に含まれるパラメータを決定することが，本方式における学習である.

ここでは，Sutton らと同様に，期待報酬値を $\rho(\pi)$ で表し，以下の様に定義する[14].

$$\rho(\pi) \equiv E[R(u)] \quad (7)$$

$$= \sum_u P(u) \cdot R(u) \quad (8)$$

ここで，軌道 u は時刻 t $(t=1,\ldots,N_u)$におけるロボットの位置 r_t の時系列であり，$P(u)$はロボットの軌道が u となる確率である. また，式(7)の右辺の記号 E[...] はその確率による期待値操作である. 期待値操作を考えるのは，次に述べるように，必ずしも目的関数 E_v の最小値（実際はある点の近傍での極小値）を与える速度ベクトルの値を決定論的に選ぶのではなく，確率的に選択するからである.

ある時刻 t において適切と思われる速度ベクトルを選択することを上では" 方策" と称したが，本方式では，方策 π として以下の Boltzmann 分布による確率的方策 (probabilistic policy)を用いる.

$$\pi(v_t; r_t, r_{t-1}, \{b_k\}) \equiv \frac{e^{-E_v(v_t)/T}}{\sum_{v_t} e^{-E_v(v_t)/T}} \quad (9)$$

ただし，$E_v(v_t)$の第2項である滑らか項は，式(5)より1ステップ前の時刻の状態に依存するので，方策 π も1ス

テップ前の時刻の状態に依存している．これにより，式(9)の方策による行動決定の過程は，非マルコフ的な行動決定過程となっている．従来より，強化学習の有力な方法としてQ学習が知られているが，Q学習の場合，行動決定の過程としてはマルコフ過程を想定している．したがって，式(5)のような非マルコフ的な知識を利用した行動計画の場合に，Q学習をそのまま適用することは難しいと考えられる．

2.4 確率的勾配法による期待報酬値の最大化

以上の準備の下で，期待報酬の $\rho(\pi)$ の勾配 $\partial/\partial b_k[\rho(\pi)]$ を計算し，確率的近似法(Robbins-Monro アルゴリズム)の一種である確率的勾配法[15]を用いると，重み係数 b_k に関する次の学習則を得る．

$$\Delta b_k = \varepsilon \cdot R(u) \cdot \sum_{t=0}^{N_u-1} \frac{\partial}{\partial b_k} \ln \pi(v_t;\{b_k\}) \qquad (10)$$

$$= -\frac{\varepsilon}{T} \cdot R(u) \cdot \sum_{t=0}^{N_u-1} \left\{ \frac{\partial E_v}{\partial b_k} - \left\langle \frac{\partial E_v}{\partial b_k} \right\rangle_{T,\{b_k\}} \right\} \qquad (11)$$

ここで，ε は学習係数とよばれる正の定数である．右辺に出てくる微分係数の値は，目的関数 E_v が式(1)のような形であれば，特に容易に計算することが可能である．式(11)の中の<...>は，各時刻における速度ベクトル v_t に関する平均操作，

$$\langle X \rangle_{T,\{b_k\}} \equiv \frac{\sum_{v_t} X \cdot e^{-E_v(v_t;r_t,r_{t-1},\{b_k\})/T}}{\sum_{v_t} e^{-E_v(v_t;r_t,r_{t-1},\{b_k\})/T}} \qquad (12)$$

を表している．この処理においても，図1のように速度ベクトルに関する探索空間を離散化し，ベクトルの大きさを限定しているので，計算量が抑えられている．

なお，式(10)の学習則は，Williams の episodic REINFORCE アルゴリズム[16]と同一である．

3 方策の空間依存性と時間依存性

本章では，政策 π の空間依存性と時間依存性とについて考察する．

3.1 走行経験から得られる局所情報の利用

2.1 の目的関数を各時刻において決定論的に最小化してしまうと，状況によってはローカルミニマムに陥り，局所的にトラップされてしまうことがある．図2にその例を示す．式(9)の確率的政策では，温度パラメータ T の値を調節することにより，この状況をある程度回避することは可能であるが，障害物の形状や配置によっては調節が難しい．

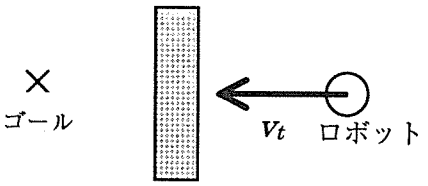


図2　局所的にトラップされてしまう場合の例

本来，式(1)の右辺で表された3つの項は，報酬の高い軌道・経路を生成するために人間が事前に与えた先見的知識を表現している．ところが，図2のようなローカルミニマムに陥りやすい場合や，迷路からの脱出問題のように，各地点においてどういう方向へ行けば目的を達成できるかという情報をそれまでの走行経験をもとに計算しておき，次に同じ地点に来たときにはその値をもとに行動を決定する方式の方が役に立つ場合がある．これを一般的に言うと，エージェントがそれまでの経験を通して得ることができた経験的知識を方策に反映させるということである．Q学習のQ値はこの経験的知識に相当すると考えられる．

このような場所に依存した経験的知識を表現するために，式(1)の目的関数 $E_v(v_t)$ に項 $E_q(v_t)$ を追加する．すなわち，

$$E_v(v_t) = b_1 E_{goal} + b_2 E_{smth} + b_3 E_{clsn} + b_q E_q \qquad (13)$$

ただし，E_q は，

$$E_q(v_t;r,v) \equiv -\sum_{r,v} q(r,v) \cdot \delta_{v,v_t} \delta_{r,r_t} \qquad (14)$$

で定義する．ここで，位置座標 r と時刻 t におけるロボットの位置 r_t とは何らかの方法で離散化されており，速度ベクトル v，v_t も図1のように離散化されているとする．δ はクロネッカーの δ 記号である．この項の意味は，位置 r において速度ベクトル v を選択したときの妥当性を，パラメータ q(r,v)で表現しておき，方策 π の学習に役立てるということである．

式(13)のパラメータ b_q や式(14)のパラメータ q(r,v)も，式(11)に示した学習則を用いて全く同様に学習することができる．また，パラメータ q(r,v)の値を格納する表のサイズが大きすぎるのであれば，(r,v)から q の値への写像を適当なパラメータ ω を含む関数で表現することも考えられる．たとえば，入力層に(r,v)を表すパターンを入力し，出力層に q(r,v)の値を出力する多階層パーセプトロン型のニューラルネットワークモデルで関数 q(r,v)を近似する方法が考えられる．

また，人間の与えた先見的知識が空間依存性を持つこ

とがわかっている場合には，式(1)や式(13)で用いた重み係数$\{b_k\}(k=1,2,3)$を空間的なパラメータ$\{b_k(r)\}(k=1,2,3)$とみなして，式(11)の学習則を適用することも可能である．これは，たとえば，障害物がない場所では，目的地への引力を表す引力項E_{goal}の重み係数b_1の値は大きい方が，障害物との斥力を表す斥力項E_{clsn}の重み係数b_3の値は小さい方が望ましく，逆に，障害物の多い場所ではこの逆の状況が望ましいと言うような場合である．

3.2 時間変動する環境下での方策学習

前節では，行動決定を行う際に，場所に依存した知識を目的関数中のパラメータの空間依存性により表現した．しかし，目的地や障害物が移動するというように，環境が時間変動する場合には，この知識も時間とともに適応的に変動させて，行動計画を立案すべきである．そこで，本節では，式(1)中のパラメータ$\{b_k\}(k=1,2,3)$を時間変動するパラメータと考えた場合の方策学習を考察する．

式(1)で表された目的関数$E_v(v_i)$のかわりに，離散時刻tを含む次の目的関数$E_{vt}(v_i;t)$を考える．

$$E_{vt}(v_i;t)=b_1(t)E_{goal}+b_2(t)E_{smth}+b_3(t)E_{clsn} \quad (15)$$

ただし，重み係数$\{b_k(t)\}(k=1,2,3)$は時間の関数であり，目的地の位置r_{goal}や障害物の位置も時間の関数とする．

このとき，式(11)の学習則は，

$$\Delta b_k(t)=-\frac{\varepsilon}{T}\left[R(l_i)\cdot\left\{\frac{\partial E_{vt}}{\partial b_k(t)}-\left\langle\frac{\partial E_{vt}}{\partial b_k(t)}\right\rangle_{T,\{b_k\}}\right\}\right] \quad (16)$$

となる．式(11)との違いは，式(11)では生成された軌道上の各点に沿って，$\{...\}$内の値を計算し，その合計を取るのに対して，式(16)では合計を取る必要がない．これは，時間依存性のない重み係数の場合，軌道上のすべての地点における行動決定に影響を与えるが，重み係数に時間依存性がある場合には，異なる時刻の重み係数は互いに独立の変数であり，他の時刻の行動決定に直接には関与しないからである．

4 マルチロボットシステムにおける行動計画

4.1 集中的な計画方式と自律分散的な計画方式

前章までは，ロボット1台の行動計画を対象としてきたが，本章では複数台のロボットシステム（マルチロボットシステム）の行動計画を立案することを考える．

マルチロボットシステムの行動計画を立案する際には，すべてのロボットの行動をひとまとめにして取り扱う集中的な計画方式（集中方式）と，個々のロボットの行動をそれぞれ個別に決定する自律分散的な計画方式（自律分散方式）とに大別される．

行動計画問題を離散最適化問題に帰着させるという立場から見ると，前者の方式では，解の精度は良くなるが探索空間が広がってしまうという欠点がある．すなわち，各時刻における速度ベクトルの組合せの個数がロボット台数のべき乗で増加してしまう．また，ロボット同士の相互作用が近接力に限られる場合だと，お互いの行動計画に影響を及ぼし合うのは，ロボット同士が接近し合った場合に限られる．したがって，毎時刻，全ロボットの行動の組合せを考える集中方式では，処理の効率が良いとは言えない．そこで，本研究では，後者の自律分散方式に基づいて，方策学習の適用を検討することにした．

4.2 ロボット同士の衝突回避と優先順位

複数台のロボットが走行するシステムにおいて，多くの場合，ロボット同士の衝突を回避する軌道・経路を計画することがしばしば要求される．この目的を実現するための一つの方法として，ロボット間に働く近距離斥力の項を目的関数に追加することが考えられる．

そこで，ロボットごとに次の目的関数を考える．

$$E_v(v_i)=b_1E_{goal}+b_2E_{smth}+b_3E_{clsn}+b_{rbts}E_{rbts} \quad (17)$$

ここで，第4項のE_{rbts}は，他のロボットとの間に働く近距離斥力で，たとえば，障害物回避のために用いた式(6)のような関数を用いれば良い．

ただし，上記の方法では，現時刻tにおけるお互いの位置座標を計画者が正確に得ており，E_{rbts}の項を通してロボット同士の衝突を回避するように，ロボット各自の行動決定に役立てている．しかし，それぞれのロボットの行動が互いに干渉し合う場合には，計画者が調停する必要が生ずる．たとえば，ロボット間の斥力項E_{rbts}の重みb_{rbts}の値が小さすぎて次時刻には接触・衝突してしまったり，お互いに進路を妨げてデッドロックに陥る場合や，逆に重みb_{rbts}の値が大きすぎてお互いに過度に避け合い，経路の最短性を損なってしまう場合などがある．

この問題は，マルチロボットシステムにおける重要な課題である．これを解決する簡便な方法の一つとして，予めロボット間に優先順位を設定しておき，計画者が優先順位の高いロボットから順に，次時刻の行動を計画するという方式が考えられる．

5 部分観測マルコフ決定過程下での行動決定

前章までは，計画者が事前に環境地図や移動障害物の時間変化に関する情報を正確に把握していることを前提としてきた．つまり，実際のロボットが走行する前に，予め計算機シミュレーションにより行動計画を立案す

るような場合を想定していた.

本章では，ロボット自身が，未知環境下で自分自身の行動決定方法（方策π）を自らの走行経験により学習することを考える．この場合，ロボットは自らの位置rを直接に知ることはできず，自らを含む環境をセンサにより得られる観測値xだけから行動決定を行う必要がある．通常，センサの観測能力は完全ではなく，かつ，確実でもないので，現時刻のロボットの位置を同定することは難しく，ロボットの行動決定過程をマルコフ決定過程で厳密に定式化することは難しい．観測が不完全・不確実なために，自らの状態を含めて環境に不確実性がある場合のエージェントの行動決定過程は，一般に，部分観測マルコフ決定過程(POMDP: Partially Observable Markov Decision Process) と呼ばれている[18]. POMDPにおいても，本アプローチが適用できることを以下に示す.

2章で述べた行動計画法を，未知環境下での各時刻における行動決定法として考える．式(1)の目的関数を計算するために必要な情報は，時刻tにおけるロボットの位置 r_t であるが，これは観測 X により推定された位置 x_t で代用する．正確な位置推定を行うには，何らかの位置推定（局在化，localization）の操作を行う必要がある[11,13,17]. 次時刻における予測位置 r'_{t+1} も $x'_{t+1}=x_t+v_t\Delta t$ で代用する．次に，斥力項 E_{obsn} の計算で使用する障害物までの最短距離値 $Dist(r'_{t+1})$は，距離センサにより得られた情報から推定する．さらに，各時刻における目的地点までの相対距離や方向が何らかの方法によりわかる場合には，引力項 E_{goal} に用いている関数 $G(v_t)$の中に反映させることも可能である.

上記の方法における目的関数中の重み係数の学習則は，式(11)に示したものと形式上は全く同じものになる．この学習則は，POMDPにおける強化学習法として，木村らが提案している確率的傾斜法の文献[18]中の定理2で報酬の割引率γの値を1とした場合の学習則になっている．ただし，本方式の場合，方策πや学習則は現時刻tにおける状態 s_t の関数ではなく，センサ値から推定された状態 s_t の推定値 x_t の関数 $\pi(v_t,x_t,\{b_k\})$であり，センサの観測値から現時刻におけるロボットの状態（位置と向き）を推定する必要がある.

また，もし，行動決定に，本方式のように目的関数の最小化という定式化を用いずに，ニューラルネットワークモデルのような関数近似法を用いて，観測Xと方策πや学習則とを直接関連付けることも可能である[16][18]. 方策πとして階層型のニューラルネットワークモデルを用いる方法は，すでにWilliamsにより提案されている[16].

6 おわりに

本報告では，先に提案した方策学習を用いた移動型ロボットの軌道・経路計画問題の解法を，いくつかの場合に拡張することを試みた.

まず，行動計画に関する先見的な知識だけではローカルミニマムに陥りやすい場合や，先見的知識以外の経験的な知識の表現方法として，空間に依存したパラメータの導入により，方策に空間依存性を持たせることを提案した．次に，移動障害物が存在する場合のように環境が時間変動する場合には，パラメータに時間依存性を持たせることにより対応する方法を考案した.

さらに，マルチロボットシステムにおいて，個々のロボットの行動計画を自律分散的に行う方式の一つとして，方策学習による本計画方式の適用可能性を示した．最後に，環境に不確実性のある部分観測マルコフ決定過程への適用について考察した．今後は，計算機シミュレーション等で本論文において提案したアルゴリズムの検証を進めて行く予定である.

謝辞

本研究に関してご討論いただいた，本学部五百井清助教授に感謝の意を表します．なお，本研究は，日本学術振興会より科学研究費補助金（C(2)，課題番号 11680405）の助成を受けた.

参考文献

[1] 柿倉正義，" 移動ロボットの経路探索，" 日本機械学会誌，第89巻，第815号，pp.1151-1156(1986).

[2] 柿倉正義，" 自律移動のロボットにおける知能化技術（1）行動プラニング"，日本ロボット学会誌，Vol.5, No.5, pp.398-402(1987).

[3] 小森谷　清，小谷内範穂，" 移動ロボットの知能"，日本ロボット学会誌，Vol.9, No.1, pp.100-111(1991).

[4] 藤村希久雄，" 行動ストラテジーとアルゴリズム"，日本ロボット学会誌，Vol.11, No.8, pp.1124-1129(1993).

[5] T.Arai, H.Ogata, and T.Suzuki, "Collision Avoidance Among Multiple Robots Using Virtual Impedance " ,IEEE/RSJ International Workshop on Intelligent Robots and Systems 9, Sep.4-6, 1989, Tsukuba, Japan, pp.479-485.

[6] D.Kortenkamp, R.P.Bonasso, and R.Murphy(eds.), Aritificial Intelligence and Mobile Robots, AAAI Press/The MIT Press(1998).

[7] Y.K. Hwang and N.Ahuja, "Gross Motion Planning -A Survey," ACM Computing Surveys, Vol.24, No. 3, pp.219-292, 1992.

[8] 小方博之，新井民夫，太田順，" 時変環境でユーザ仕様を考慮した移動ロボットの軌道計画法，" 日本ロボ

ット学会誌，Vol.12，No.6，pp.905-910(1994).
[9] B.H.Krogh and C.E. Thorpe, " Integrated Path Planning and Dynamic Steering Control for Autonomous Vehicles," Proc. of IEEE Int. Conf. on Robotics and Automation, pp.1664-1669(1986).
[10] J.Latombe, Robot Motion Planning, Kluwer Academic Publishers(1991).
[11] 五十嵐治一，五百井清，" 最適化問題としての経路計画と走行誘導"，第 4 回ロボティクスシンポジア予稿集 pp.269-274(1999).
[12] 五十嵐治一，" 強化学習を用いた自律移動型ロボットの経路計画，" 第 5 回ロボティクスシンポジア予稿集，pp.403-408(2000).
[13] 五十嵐治一，" 離散最適化問題としての走行誘導・経路計画と強化学習によるパラメータ決定法"，人工知能学会第 6 回 SIG-Challenge 研究会資料，pp.7-12('00, 6 月，函館)
[14] R.S.Sutton, D.McAllester, S.Singh, and Y.Mansour, "Policy Gradient Methods for Reinforcement Learning with Function Approximation", Advances in Neural Information Processing Systems 12 (NIPS12), pp.1057- 1063(2000).
[15] 石井健一郎，上田修功，前田英作，村瀬洋，パターン認識，オーム社，pp.166-171(1998).
[16] R.J.Williams, "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning", Machine Learning, 8, 229-256(1992).
[17] S.Thrun, A.Bucken, W.Burgard, et al.,"Map Learning and High-Speed Navigation in RHINO," in [6], pp.21-52.
[18] 木村元，山村雅幸，小林重信，" 部分観測マルコフ決定過程下での強化学習：確率的傾斜法による接近"，人工知能学会誌，Vo.11, No.5, pp.85-92.

対戦型マルチエージェントシステムにおける チーム構成の戦略の獲得

Strategy Learning For A Team In Adversary Environments

田村 隆 高橋 泰岳 浅田 稔

Takashi TAMURA, Yasutake TAKAHASHI, and Minoru ASADA

大阪大学大学院 工学研究科 知能・機能創成工学専攻

Dept. of Adaptive Machine Systems, Graduate School of Engineering Osaka University

{tamtam,yasutake,asada}@er.ams.eng.osaka-u.ac.jp

Abstract

Team strategy acquisition is one of the most important issues of multi-agent systems, especially in an adversary environment. RoboCup has been providing such an environment for AI and robotics researchers. A deliberative approach to the team strategy acquisition seems useless in such a dynamic and hostile environment. This paper presents a learning method to acquire team strategy from a viewpoint of coach who can change a combination of players each of which has a fixed policy. Assuming that the opponent has the same choice for the team strategy but keeps the fixed strategy during one match, the system estimates the opponent team strategy (player's combination) based on game progress (obtained and lost goals) and notification of the opponent strategy just after each match. The trade-off between exploration and exploitation is handled by considering how correct the expectation is. A case of 2 to 2 game was simulated and the final result (a class of the strongest combinations) was applied to RoboCup-2000 competition.

1 Introduction

チーム戦略の獲得はマルチエージェントシステムにおいて，特に敵対するエージェントが環境に存在するとき非常に重要な問題の一つである．ロボカップではAIとロボット研究者のためにこのような環境を提供してきた．この環境では従来から研究されてきたマルチエージェント環境と比べ環境の変動が速く実時間処理が必要とされ，さらに敵の存在のためモデルや計画を立てることを困難にしている．そのため環境の情報を直接使ったフィードバックを使った制御が有効である[1].

シミュレーションリーグではさまざまなチーム戦略の獲得の手法が提案されて来た．一方で実機リーグではハードウェアやシステム全体の構成の方に重点がおかれ，それほど多くの研究がチーム戦略の獲得に関して行われて来たわけではない．Castelpietra et al. [2]は無線を通して情報や役割を交換することにより協調動作を創発する手法を提案した．Uchibe et al. [3]はタスクの達成度を共有メモリに書き込むことで動的に役割分担させる枠組を提案した．しかしこれらの手法はあらかじめチーム戦略を固定していると考えられる．

本論文ではエージェントがそれぞれの個性を持った固定政策をもち，エージェントの組合せを決定できるコーチの視点からチーム戦略を獲得する学習方法を提案する．敵チームも同種類のチーム戦略の選択権があり，ゲーム中は戦略を固定している．学習システムはゲーム中敵チームの戦略を予測し最適なチーム戦略を選択しながら戦い，試合後に敵チームの戦略を教えてもらうのとする．学習中の探索するための戦略か最適な戦略をとるかのトレードオフは試合結果が予測とあっているかによって自動的に決定する．2対2の試合におけるシミュレーションを通して本手法の有効性を示す．

2 問題設定

環境がより複雑になるにつれ，システムはそれに対応するためにより繁雑になっていく傾向にある．特にマルチエージェントシステムでは協調動作を実現するためにコミュニケーションは有効なようにみえる．しかしながら環境が非常に動的で，それゆえに環境の変化の予測が難しいような環境では役に立たない．行為に基づく手法はこのような動的に変化する環境に対応できる一つの方法である．しかし対戦型マルチエージェント環境においてそれぞれのロボットを設計することは相手の戦略を前もって

知ることができないので困難である．

そこで我々は以下の条件のもとでチーム戦略を獲得する学習手法を提案する．

- それぞれのチームは多様性を持った選手を擁し，それぞれの選手は互いにコミュニケーションせず，ビヘービアベースドアプローチによって設計されている．
- チーム戦略は選手の組合せによって定義される．
- 監督は選手を交替させることでチーム戦略を変える．
- 相手チームも同じ能力を持った選手を擁し，同じ戦略をとることができる．

ゲームの進行状況から監督の視点で相手のチームに勝てる戦略を発見することが目的となる．したがって監督は相手チームの戦略を試合経過を通して推定し，自分のチーム戦略との関係を獲得していく必要がある．そこで以下の仮定をおく．

- 相手チームは毎試合ごとに戦略をランダムに変更するが，試合中は変更しない．一方自チームの監督は試合中に何度か戦略を変更することができる．

これ以降，監督とは学習するチームの監督を指すものとする．

3 チーム戦略学習

まず学習するチームの監督にとって二つの政策を用意する．

チーム探索政策 得失点表を作るための探索行動

最適チーム推定政策 相手のチームを推定し，勝てるチームを算出

チームの選択政策はチーム探索政策と最適チーム推定政策の重みつきの足し合わせで決定する．その重み w_{er} はチーム探索政策と最適チーム推定政策の比率を決めており，最適チーム推定政策の推定結果が適切になるにつれてチーム探索政策から最適チーム推定政策へ移行していくように重み w_{er} を更新する．j 番目の戦略をとる確率 $P(j)$ を以下のように定義する．

$$P(j) = (1 - w_{er})P_r(j) + w_{er}P_e(j) \tag{1}$$

ここで $P_r(j)$ と $P_e(j)$ はそれぞれチーム探索政策と最適チーム推定政策により決定される j 番目の戦略をとる確率である．

3.1 チーム探索政策

チーム探索政策は非常に単純で，最も経験の少ない戦略を選択する政策である．監督は一試合の中で何回か戦略を変更する機会を持つ．ここでは戦略を変更するまでの

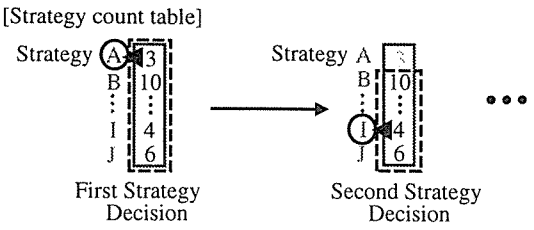


Figure 1: Exploration policy

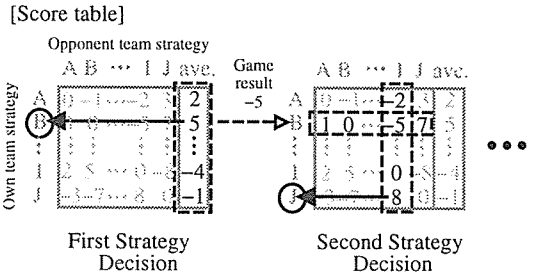


Figure 2: Exploitation policy

期間をピリオドと呼ぶ．システムは n 個ある戦略のうち j 番目の戦略をとった回数を保存した試合回数表をもつ．チーム探索政策により次のピリオドで j 番目の戦略を選択する確率 $P_r(j)$ を以下のように定義する．

$$P_r(j) = \frac{p_r(j)}{\sum_{l=1}^{n} p_r(l)} \tag{2}$$

ただし

$$p_r(j) = \begin{cases} 0 & (j \text{ 番目の戦略を既に行った}) \\ 1 & (\text{if } T_r(j) = 0\ ) \\ \dfrac{1}{T_r(j)} & (\text{それ以外}) \end{cases} \tag{3}$$

Figure 1 に式 (2) と (3) の考え方を示す．監督は第一ピリオドでは試合回数表から経験の少ない戦略を選択する．第二ピリオド以降では既にとった戦略以外の中から経験の少ない戦略する．

3.2 最適チーム推定政策

最適チーム推定政策は相手チームの戦略を推定する手続きと推定した相手に勝てる戦略を推定する手続きからなる．k 番目の戦略を相手がとり j 番目の戦略を自チームがとったときの得失点差を記録する得失点表 $T_s(k,j)$ をシステムが保持している．自チームが第 i ピリオドに j 番目の戦略をとり得失点 s_j^i を得たとき，システムは相手チームの戦略を以下のように推定する．

$$P_s^i(k) = \frac{p_s^i(k)}{\sum_{l=0}^{n} p_s^i(l)} \tag{4}$$

$$p_s^i(k) = \max_l |s_j^i - T_s(l,j)| - |s_j^i - T_s(k,j)| \tag{5}$$

(5) 式右辺の第一項は推定の最大誤差を返す．第二項は相手の戦略が k であると推定したときの誤差であり，推定が正しければ小さな値を返す．したがって $p_s^i(k)$ は相手の戦略が k 番目であれば大きな値を返す関数になるので，これを式 (4) で正規化することで相手の戦略が k 番目である疑似的な確からしさを算出する．あるゲーム中に m ピリオド経過した場合，相手チームのとっている戦略は次式で推定する．

$$P_s(k) = \frac{1}{m}\sum_{i=0}^{m} P_s^i(k) \tag{6}$$

次に監督が j 番目の戦略をとったときに得られる得失点差の推定値 $x_s(j)$ を以下で計算する．

$$x_s(j) = \sum_{k=1}^{n} P_s(k)T_s(k,j) \tag{7}$$

これから最適チーム推定政策として j 番目の戦略をとる確率を次のように定義する．

$$P_e(j) = \frac{p_e(j)}{\sum_{l=0}^{n} p_e(l)} \tag{8}$$

where

$$p_e(j) = \begin{cases} 0 & (\text{if } x_s(j) \leq 0) \\ x_s(j) & (\text{else}) \end{cases} \tag{9}$$

Figure 2 に最適チーム推定政策の基本的な考え方を示す．チームの監督はまず得失点表から平均的に強いチームを割り出し第一ピリオドに臨む．(ここでは最大の平均値 5 を取った B 戦略を取る．) 次のピリオドが来たら得失点表から相手チームの戦略を割りだす．(直前のピリオドに取った戦略 B と得失点差-5 から相手チームは I 戦略を取っていると予想する．) その戦略に勝てる戦略を推定し実行に移す．(相手が I 戦略を取っているものと仮定して J 戦略をとる．)

3.3 重み w_{er} の更新

一試合は n_p ピリオドにわかれており，第 i ピリオドに a^i 番目の戦略をとった時，チーム探索政策と最適チーム推定政策の割合を決定する重み w_{er} を以下のように更新する．

$$w_{er} \leftarrow w_{er} + \Delta w_{er} \tag{10}$$

$$\Delta w_{er} = \alpha \sum_{i=2}^{n_p} s^i \frac{i}{n_p} x_s(a^i) \tag{11}$$

ここで α は更新率である．もし重み w_{er} が 1 より大きくなった場合は 1 に，0 より小さくなった場合は 0 に変更する．この式の意図は重み w_{er} を学習初期はチーム探索政策を優先し，推定がうまくいくようになれば最適チーム推定政策を優先させるように自動的に更新させることにある．式 (11) は推定が正しければ w_{er} を増やす形になっている．$\frac{i}{n_p}$ はデータが少ないときの予想よりも，数ピリオド戦った後のデータが豊富な時の予想を大きく考慮させるための項である．

4 実験

4.1 設定

ここでは以下の仮定をおく．

- 選手とフィールドは RoboCup2000 の規定に則している．
- 各チームは 2 選手を擁する．
- 選手はそれぞれ個性を持った固定政策で動く．
- 各選手はコミュニケーション能力を持たない．
- 一試合は 500 試行である．
- 100 試行を 1 ピリオドとする．つまり一試合中 5 回の戦略変更が可能である．
- 選手の初期配置は自陣の中でランダムであり，ボールは中央におかれる．

4.2 チーム戦略

ここではチーム戦略は監督の立場で出場させる選手の組合せを決めることとする．4 種類の選手を用意した．まずボールに対してアプローチする制御器を二種類用意する．

ROUGH ボールへのアプローチが速く，ハンドリングは荒い．

CAREFUL ボールへのアプローチハンドリングともに慎重．

つぎに見方を認識したときの振舞として次の二種類を用意する．

SELFISH 敵も味方も障害物としてのみ認識する．

SOCIAL 味方を識別し，味方に道を譲る．

この二つを組み合わせることで 4 種類の制御器がある．チーム戦略としては二選手の組合せ方で 10 種類ある．

4.3 実験結果

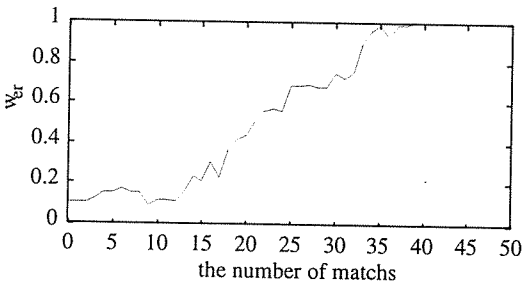


Figure 3: The change of w_{er}

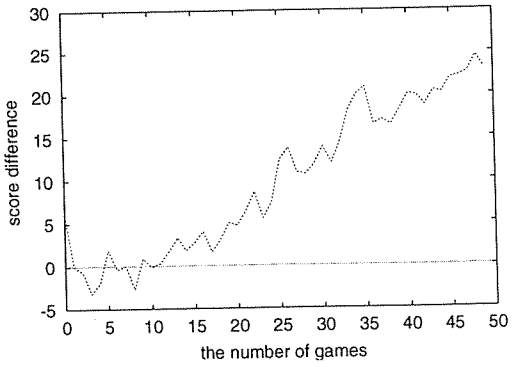


Figure 4: Difference of obtained and lost goals

Figure 3は，試合回数に伴ってチームの選択政策がチーム探索政策から最適チーム推定政策に移行していることを示すグラフである．横軸が試合回数を示し，縦軸は重み w_{er} を表す．Figure 4 はチームの選択政策がチーム探索政策から最適チーム推定政策に移行するにしたがって，経験に基づき試合を有利に進めていることを示すグラフである．Figure 3の重み w_{er} の推移と比較すると，重み w_{er} が大きくなり最適チーム推定政策になるにつれ，大きく得点差をつけ試合に勝っていることが分かる．

Table 1: The result of differnece of obtained and lost goals by learning

own	opponent										ave.
	1	2	3	4	5	6	7	8	9	10	
1	-	2.3	-4.2	5.5	5.7	6.3	3.0	-	3.2	11.6	4.2
2	-2.3	-	1.2	1.1	6.4	4.1	11.3	-	-	8.3	4.4
3	4.2	-1.2	-	-	-	-3.4	-	-	-3.8	-	-0.6
4	-5.5	-1.1	-	-	-	-	-	-	-	-	-2.8
5	-5.7	-6.4	-	-	-	-	-	-	-	-	-4.1
6	-6.3	-4.1	3.4	-	-	-	-	-	-	-	-2.9
7	-3.0	-11.3	-	-	-	-	-	-	-	-	-4.3
8	-	-	-	-	-	-	-	-	1.1	-	-1.3
9	-3.2	-	3.8	-	-	-	-	-1.1	-	-	-1.0
10	-11.6	-8.3	-	-	-	-	-	-	-	-	-4.3
ave.	-4.2	-4.4	0.6	2.8	4.1	2.9	4.3	1.3	1.0	4.3	-

1:(RO-SE,CA-SO) , 2:(RO-SE,CA-SE) , 3:(RO-SE,RO-SO) , 4:(CA-SE,CA-SO) , 5:(RO-SO,CA-SE) , 6(CA-SE,CA-SE): , 7:(RO-SO,CA-SO) , 8:(RO-SE,RO-SE) , 9:(RO-SO,RO-SO) , 10:(CA-SO,CA-SO)
RO:ROUGH , CA:CAREFUL,
SE:SELFISH, SO:SOCIAL

Table1は学習が終わったと見なせる時点 (50 試合) で 5 ピリオド (1 試合分) 以上の試合が行われた組合せの結果だけを選んだ対戦成績である．ただし表内の値は 1 ピリオドでの得失点の平均である．探索範囲が上位チームを中心に探索されている．このことにより総当たり戦より狭い範囲を重点的に探索していることが分かる．総当たりするよりも約 1/3 の時間ですんだ．

Figure 5は Table1の結果を用いて作成したチーム間の相性の図である．サイクル状の関係に成っていることから，一番強いチームを選んで来てそれを実行しさえすれば良いことにならないことがわかる．下位のチームの相関関係が分からなくても試合に勝てる上位のチームを獲得しているため，試合には勝つことができ支障はない．また総当たり戦よりも少ない情報 (試合数) で敵チームに勝

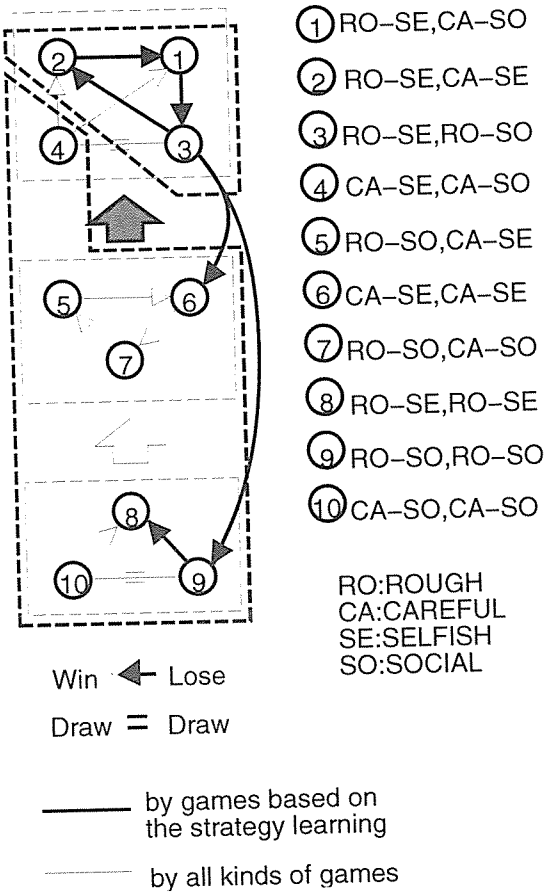


Figure 5: The learned relationship among strategies

てるチーム構成の戦略が獲得できている．以上より**本手法の探索は少ない探索で必要十分な戦略を獲得できる**ことを示せた．

次に 2000 年 7 月にオーストラリアのメルボルンで行われた RoboCup2000 の試合中に，創発された協調行動と異種混合のチーム構成の戦略による効果について示す．この試合で用いているコントローラーの種類は以下の 2 種類である．

- typeA : CAREFUL-SELFISH
- typeB : ROUGH-SOCIAL

Figure 6は 2 台のロボットがお互いの失敗をフォローしている様子を示している．①は 2 台のロボットがボールへ向かっている様子である．②では typeB のロボットが敵のゴールへシュートを試みている．③では typeB のロボットがシュートを失敗し，それを typeA のロボットがカバーしている．④では typeA のロボットがシュートを試みたが敵のゴールキーパーに阻止されている．⑤，⑥を通して typeA のロボットがゴールの左側からシュートを試みたが再び失敗している．この時に typeA のロボットの後ろにいた typeB のロボットが⑦ において typeA のロボットのカバーをしシュートを試み，⑧においてシュートを成功させている．

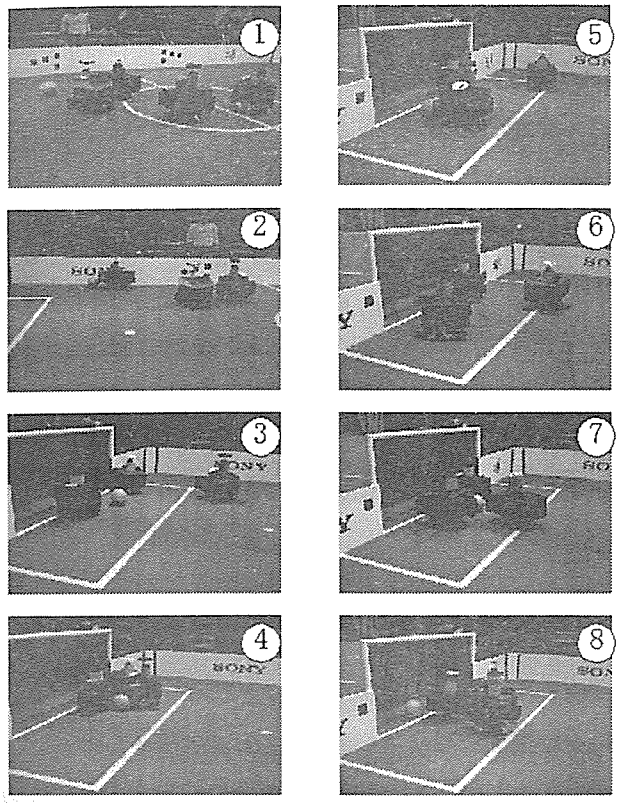


Figure 6: A sequence of a failure recovery behavior among two robots

5 Conclusions

本論文ではエージェントがそれぞれの個性を持った固定政策をもち，エージェントの組合せを決定できるコーチの視点からチーム戦略を獲得する学習方法を提案した．2対2の試合におけるシミュレーションを通して本手法の有効性を示した．

参考文献

[1] Barry Werger. Cooperation without deliberation: Minimalism, stateless, and tolerance in multi-robot teams. *Artificial Intelligence*, 77:293–320, 1999.

[2] Claudio Castelpietra, Luca Iocchi, Daniele Nardi, Maurizio Piaggio, Alessandro Scalso, and Antonio Sgorbissa. Communication and coordination among heterogeneous mid-size players: Art99. In *The Fourth International Workshop on RoboCup*, pages 149–158, 2000.

[3] Eiji Uchibe, Tatsunori Kato, Minoru Asada, and Koh Hosoda. Dynamic Task Assignment in a Multiagent/Multitask Environment based on Module Conflict Resolution. In *Proc. of IEEE International Conference on Robotics and Automation*, 2001 (to appear).

PVM を用いたロボットのタスクの分散化に関する検討

A Study on Distribution of Tasks of Robots utilizing PVM

中野博史、鈴木秀智、長内真人、安井慎太郎、渡辺一郎、樫尾次郎
Hiroshi NAKANO, Hidetomo SUZUKI, Masahito OSANAI,
Shintaro YASUI, Ichiro Watanabe, Jiro KASHIO
三重大学工学部
Faculty of Engineering, Mie University
{nakano,suzuki,osanai,yasui,wata,kashio}@kashio.info.mie-u.ac.jp

Abstract

This paper describes the system configuration of our robots, and a method of distribution of tasks on robots with PVM. PVM is a tool to integrate many computers connected through network as a distributed parallel computer and execute parallel processing. PVM can handle the task running on robots with the same manner by use of the message passing mechanism. This makes easy to develop and maintain the robots' programs. An experiment shows that the latency of multi-thread programming and multi-process programming is nearly negligle for controlling a robot.

1 はじめに

人工知能と知能ロボットに関する研究のひとつの標準問題として、ロボットによるサッカー競技 RoboCup が提唱され、種々の技術の統合が必要な複合的問題として注目されている。著者らの研究室では、これまで画像処理、ネットワーク通信技術およびその応用であるマルチメディア処理に関する研究を行ってきており、これらの研究成果の応用およびその知的処理への展開のための題材として RoboCup を採用した。主な研究対象は、自律移動ロボットの視覚情報処理の精度向上、情報の共有による情報補完、行動計画の高度化、協調動作の実現であるので中型リーグを選択した。これらの問題については既に多くの研究グループによって成果が出されているが、解決すべき問題はまだ多く残されていると考える。

本報告では、上記の問題を扱うために製作したロボットシステムの概要を紹介し、PVM (Parallel Virtual Machine) [Geist, 1994] を利用したロボットシステムの制御プログラムの開発について述べる。さらに、PVM を利用した場合の応答遅延は、ある一定条件下では大きな問題にならないことをを実験により示す。

2 ハードウェア構成

ロボットは、大別すると制御部と移動機構部からなる (Fig. 1)。おおまかな構成は大阪大学 Trackies の自律移動ロボットを参考にさせていただいた。

2.1 制御部

ロボットの中枢であり、IBM AT 互換の CPU カード (PCI-371,JDS) を 1 個搭載している。CPU は Intel 社の Celeron(433MHz) であり、64MB の主メモリを搭載している。2 個のシリアルポートは、ビデオカメラの制御および移動機構部との通信に使われている。パラレルポートは手動操作時の指示に用いている。Ethernet ポートを有し、無線 LAN 装置を介して複数ロボットで構成される LAN に接続されている (LAN Anywhere, Callus. or WLI-PCI-L11, Melco)。補助記憶装置として IDE 接続のハードディスク (15GB) を用いている。

最も負荷の重い画像処理は下記の画像処理ボードに担当させ、移動に関する詳細な制御は移動機構部に任せている。これにより、この CPU カードの貴重な計算資源を環境の認識、他のロボットとの情報交換、およびそれらに基づく行動計画と実行（移動機構部への指令を含む）に割り振ることが可能になった。

ビデオカメラ (EVI-D30,Sony) は、テレビ会議用の高機能のものであり、首振り機構、自動焦点機能、対象物追跡機能などが可能である。ただし、本研究では、対象物追跡機能は使用せず、独自の対象物探索および追跡機能を開発した。

画像処理ボード (IP5005BD, 日立) は、ビデオカメラから送られてくる映像を高速に処理するための専用プロセッサを搭載したものであり、映像から各種の対象物（ボール、ゴール、ロボットなど）を認識するために色情報を用

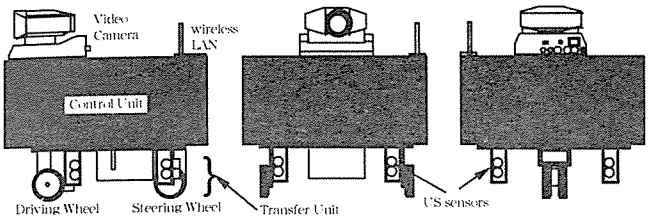


Figure 1: Overview of our robot

いた画像処理手法を実行する。現在、1 秒間に 15 フレーム程度の処理が可能である。

電源は密封型鉛蓄電池を用い、交直両用の電源装置 (NSP2-180-H2X, 日本プロテクタ) を介して必要な直流電圧を発生させている。

2.2 移動機構部

ロボットの運動を担当する部分であり、既製の移動用台車 (TRIPTERS-mini, JSD) を用いている。2 輪駆動 1 操舵輪方式の台車であり、両駆動輪および操舵輪にロータリーエンコーダが付いていて、駆動輪の移動距離および操舵輪の回転角を知ることができる。この台車には V25 CPU ボード、モータ制御ボード、超音波センサ (8 方向) が搭載されているので、この台車のみで障害物を検知し、それに応じて行動することも可能である。また、RS-232C インターフェースをもっており、これを介して、ホスト側とプログラムやデータを送受したり、搭載された OS への操作指示を受け取ったりすることができる。

本研究では、上記の仕様を活かして、移動に必要な制御機能 (基本動作、動作の達成度の検査、動作の中断、再開など) を移動機構部で実現し、制御部の移動制御を単純化した。なお、超音波センサの距離情報は不可視部分の認識や衝突回避に利用されている。

3 ソフトウェア構成

3.1 制御部

制御部で使用している主なソフトウェアを以下に示す。

- OS : Linux, Version 2.0.36　(Vine Linux 1.1)
- プログラミング言語 : C++ (g++)
- その他の開発環境 : GNU ソフトウェア (emacs, vi, gdb, etc.)
- 分散並列処理環境 : PVM

制御部では以下のタスクを実時間応答が可能な速度で実行しなければならない (Fig. 2)。なお、ここでは、PVM の用語法に基づき、"タスク"は仮想並列コンピュータ上で動作しているプログラム単位 (プロセス) を表すものとする。

(1) LAN : 無線 LAN を介して外部と通信する。

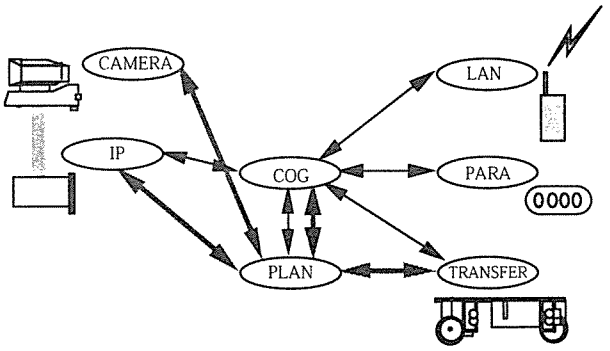


Figure 2: Configuration of tasks

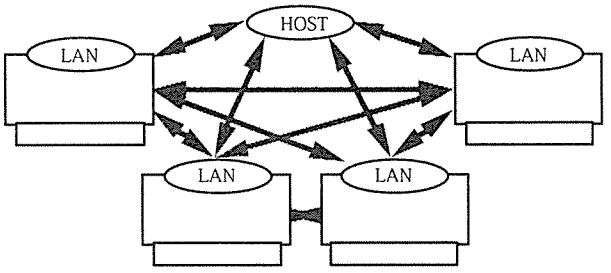


Figure 3: Communication between robots

(2) COG : 各種の情報を収集して、ロボットが置かれた環境を認識する。
(3) PLAN : 認識結果に基づいて行動を計画し、各タスクに指令を送る。
(4) TRANSFER : 移動機構部と通信する (指示、応答)。
(5) CAMERA : ビデオカメラと通信する (指示、応答)。
(6) IP : 画像処理ボードを制御し、処理結果を取得し、蓄積する。
(7) PARA : CPU カードのパラレルポートを制御する (手動操作用)。

本研究では、仮想並列コンピュータを実現するソフトウェア PVM を利用して、各タスクを分散処理する方法を採用した。これにより、メッセージパッシングによるオブジェクト指向的な各タスクの並列処理が可能になる。タスクは固有のデータを管理し、他のタスクからの情報要求メッセージに対して必要な部分のみを通知するので、データの一貫性や独立性を実現できる。したがって、各タスクを個別に扱うことができ、ソフトウェアの生産性および保守性が向上する。

また、ロボット全体に起動や停止などを指示するホストコンピュータ、および、各ロボットの LAN タスクを PVM で実現することにより、チーム全体をひとつの分散処理システムとして一貫した形態で扱うことが可能になる (Fig. 3)。

認識を担当する COG と、行動計画と実行を担当する PLAN を分けた理由は、シミュレーションリーグでの協調動作に関する研究成果を有効利用するためである。サッ

カーサーバが COG に対応し、エージェントが PLAN に対応すると考え、PLAN のインターフェースをエージェントのそれに合わせることにより、シミュレーションリーグのエージェントをロボットに容易に取り込むことができる。もちろん、エージェントが獲得した戦略などの知識をそのまま本ロボットに適用することは難しいので、この点に関する詳細な検討が必要である。

3.2 移動機構部

移動用台車には MS-DOS の縮小版に相当する JS-DOS と呼ばれる OS が搭載されている。したがって、ソフトウェア開発手順は、MS-DOS を利用したものとほぼ同じである。具体的には、MS-DOS の実行形式プログラムを作成し、それを台車の CPU ボードに搭載された EEPROM にロードして実行する。

本研究で作成した制御プログラムは、制御部の TRANSFER タスクから受け取った行動指令の解読と実行、および、超音波センサの距離情報の取得と TRANSFER タスクへの通知を行う。TRANFER タスクから受け取る行動指令は、距離や速度の指定が可能な前後進や旋回に関する指令である。それらの指令を実行するときは、移動距離や速度の検出、それに基づく目標達成のための制御、モータ過負荷時の対応など複雑な処理が必要である。このような処理を行うために、台車に添付されていた制御用ライブラリを本目的に合うように変更し、それを利用した。

4 PVM を用いたタスクの分散化

4.1 PVM (Parallel Virtual Machine)

PVM は、メッセージパッシング機構を用いた移植性の高いプログラミングシステムであり、これを用いると複数のコンピュータをひとつの仮想並列コンピュータとして扱うことができる。この仮想並列コンピュータを構成するコンピュータは互いに離れたところに設置された異機種のコンピュータであってもよい。PVM アプリケーションを構成する各タスクは、C、C++、Fortran などで書かれる。現在、PC、ワークステーション、マルチプロセッサ、スーパーコンピュータなど数多くのコンピュータで利用できる。PVM はいくつかの機関で共同で研究開発が進められていて、GPL ライセンスに基づくフリーソフトとして公開されている。たとえば、Netlib [Netlib, http] からもダウンロードできる (日本国内にミラーサイトあり)。

ネットワーク接続されたコンピュータを用いて分散並列処理をするときは、ソケットライブラリによるネットワーク通信を行う必要があるが、接続するマシンの管理、負荷の分散、通信制御などネットワークを意識した処理が必要になり、これが分散並列処理プログラムを作成するときの障害になっている。PVM は、ネットワーク通信および select システムコールによる非同期通信 (通信の多重化) の繁雑な処理を受け持ってくれるので、PVM を利用した分散並列処理プログラムの作成は容易である。

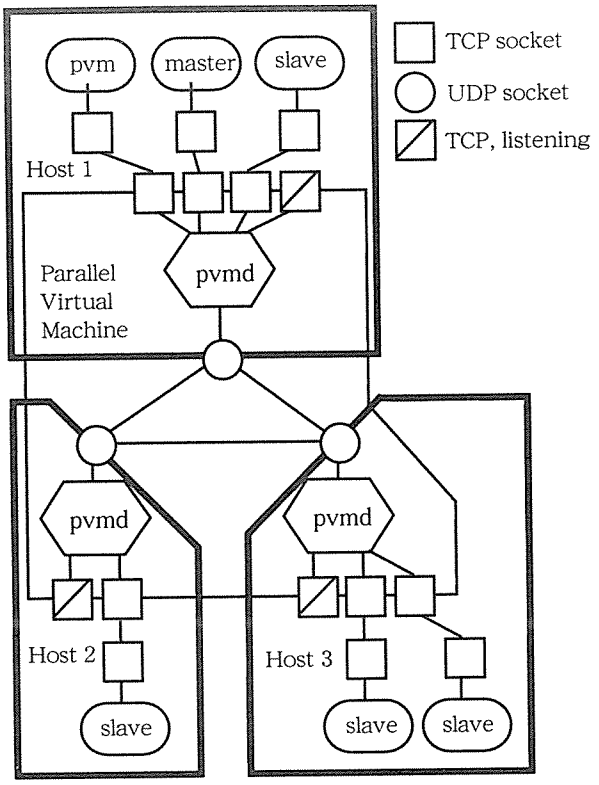


Figure 4: Overview of PVM system

PVM の構成を Fig. 4 に示す。仮想コンピュータを構成するホスト (コンピュータ、プロセッサ) に pvmd デーモンが起動され、これらが相互通信することによって仮想並列コンピュータが構築される。タスク (並列動作するプログラム) 間の通信は pvmd を介して TCP ソケットを通して行われる。pvmd 間の通信は UDP ソケットによって行われる。

ユーザは、まず、マスタープログラムを起動し、並列動作させるプログラムをスレーブプログラムとして仮想並列コンピュータ上で複数起動することによって並列処理を実行する。タスク間の通信は、タスク固有の ID およびメッセージ ID をメッセージに添付して行う。各ホストの物理的な配置および通信方法は PVM によって行われるので、ユーザは各ホストについて考慮する必要はない。もちろん、ホストを指定した配置も可能である。通信方法には、同期 (ブロック) 方式と非同期 (ノンブロック) 方式があり、必要に応じて使い分けることができる。

PVM はネットワーク接続された異種コンピュータ間のデータ交換を扱うので、ネットワークを介したデータ交換の時間、データ表現の変換の時間など余分な処理時間を要する。したがって、単一プロセッサ内で実行したり、同一アーキテクチャのコンピュータ間で通信したりするのに比べると実行速度が低下するという欠点がある。PVM にはこの問題を軽減するためにメッセージバッファによ

る通信制御などが採用されている。この種の遅延がロボットの制御にどの程度影響するかは重要な問題であるので、後の節で検討する。

4.2 他の並列処理用ソフトウェアとの比較

並列処理を実現するソフトウェアは多く存在し、専用言語およびライブラリに大別される。

4.2.1 専用言語

専用言語としては、 Concurrent {C, C++, Pascal, Euclid}、Modula-{2,3}、Ada などのように、並列処理に必要な基本機能であるプロセスの生成や削除などを文法仕様としてもつものがある。プロセスのプロセッサへの配分などは言語処理系が行うので、ユーザはこの作業をしなくてもよいが、処理の分割や並列化については指定する必要がある。

スーパーコンピュータなどでよく利用される HPF(High Performance Fortran) [HPF, http] は Fortran をベースにして並列化のための指示行を追加したものである (HPC++ という C++をベースにしたものもある [HPCPP, http])。この言語処理系は MPI などを利用した SPMD (Single Program Multibple Data Stream) 形式のプログラムに変換するものが多い。ユーザはプログラム内で並列化したい部分を指示行で指定するだけでよく、プログラムの SPMD 化、通信管理などは処理系によって実現されるという特長をもつ。また、共有メモリモデル用として標準化されたコンパイラ指示行ベースの API として OpenMP [OpenMP, http] がある。

4.2.2 ライブラリ

MPI (Message-Passing Interface) は MPI フォーラム [MPI, http] という任意参加の会議で策定された仕様であり、実際のソフトウェアを指すものではないが、この仕様に準拠したソフトウェアやライブラリを指すのに使われることが多い。MPI の実現 [MPIIMP, http] は数多くあり、並列マシンメーカが提供するものや、MPICH、MP-MPICH、LAM/MPI などのフリーのものがある。MPI は PVM より後に仕様策定されたものであり、PVM などの主要なメッセージパッシングライブラリの開発者や並列計算機ベンダーがこのフォーラムに参加しているので事実上の標準になっている。しかし、元々は、MPP (Massively Parallel Processor) に実装されていたため、同種プロセッサ向けであり、プログラムの実行に使用するプロセッサ数やプロセス配置が実行開始時に決めなければならず、SPMD 形式しか扱えない、という制約があった。現在の仕様である MPI-2 では、これらの制限が緩和されているが、実際の MPI ソフトウェアが完全に対応しているとは言えない。MPI が標準仕様になりつつあるので、実装の方が安定すれば PVM から移行することを考えている。基本的には同じ方式の並列化ライブラリなので移行はそれほど難しくない。

マルチスレッドとは、ひとつのプロセス内で複数の処理の流れ (スレッド) が生成され実行されているものである。この方式では、各スレッドはデータを共有し、コンテクストスイッチにおけるオーバヘッドが非常に少ない、などの特徴をもつので、計算機資源にあまり負荷をかけずに並列処理を実現できる。ただし、共有データの排他制御はユーザの責任となる。この方式を実現するライブラリとしては、POSIX (Portable Operating System Interface) に準拠した pthread、LinuxThread [LinuxThread, http] などがある。

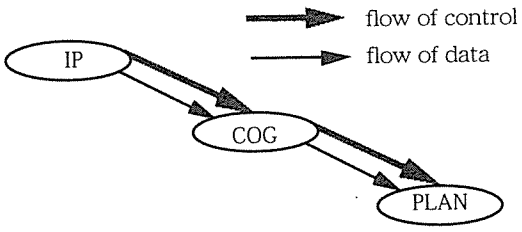
その他、ソケットライブラリ、select などの I/O 同期ライブラリ、共有メモリライブラリを用いた従来の並列処理プログラム作成法がある。周知のように、この方法では、並列処理に関する煩雑な処理 (プログラムの分割、通信管理など) をユーザが行わなければならず、ソフトウェア開発の効率は良いとは言えない。

上記の言語およびライブラリはそれぞれに特色があるので、いずれがロボット制御プログラムの分散並列化に向いているかは一概に言えない。ただ、ソフトウェア開発の効率化、拡張性、などを考えると、柔軟な対応や拡張が可能なライブラリの方が好ましいと考える。これは、本研究で PVM を採用した理由のひとつである。

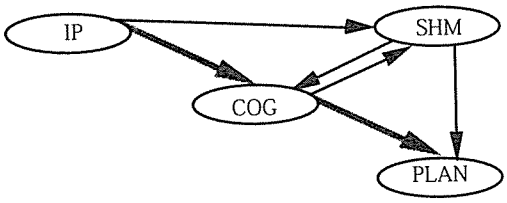
4.3 応答の遅延

ロボットを制御するプログラムを作成する上で最も重要な問題は、環境の認識結果に対する行動の遅延である。この遅延時間は前節で紹介したプログラミング手法に大きく依存する。ロボットを制御するプログラムを並列動作可能な部分に分割して、それらを並列動作できるようにプログラミングする場合、(a) OS によるプロセスまたはタスクのスケジューリング、(b) データ通信の速度、(c) プロセスまたはタスクのコンテクストスイッチのオーバヘッドなどが実時間応答性に影響すると考えられる。(a) については、リアルタイム OS を採用するのが確実であるが、ひとつのプロセッサの負荷を抑えることで実用範囲に収めることが可能である。(b) にはプロセス間通信とネットワーク通信があるが、前者は OS に依存し、後者はネットワークの通信速度や込み具合に依存するので、ユーザ側で対処することは難しい。(c) についてはマルチスレッドプログラミングを用いるのが好ましい。

森ら [森, 2000] の研究など、最近のロボット制御プログラムの多くはマルチスレッドプログラミングを用いている。確かに、マルチスレッドプログラミングは軽量な並列処理プログラムを作成するのに向いている。しかし、プログラムが大規模になったときの共有データの排他制御の問題、および、外部プロセスとの通信のためのプログ



(a) multi-thread, PVM



(b) shared memory

Figure 5: Tasks used in experiments

ラミングの煩雑さが、制御プログラムの拡張や保守において問題となる。

共有メモリなどを用いたマルチプロセスプログラミングでは、コンテクストスイッチのオーバヘッド、プロセス間通信制御のためのプログラミングの煩雑さなどが問題になる。

一方、PVM (または MPI) を用いた並列処理プログラムの場合は、コンテクストスイッチやライブラリ独自の処理によるオーバヘッドが大きくなるが、プログラミングの効率は高くなると考えられる。なぜなら、プロセス (タスク) の生成と削除、プロセス間通信、各種排他制御について、ユーザ側では最低限の指示ですむように設計されていて、プロセッサ内とプロセッサ間の通信を同じ形式で行えるからである。このことは、ロボット制御プログラムの開発の効率を高めるのに役立つ。

以上の考察において、(c) がプログラミング手法によってどの程度影響するのかを実験で確かめる。

自律移動ロボットを用いて、Fig. 5に示すタスクを以下のような方法で実行したときの処理時間を計測した。なお、今回は、プロセッサ間の通信を考慮した実験を行っていない。これは、プロセッサ間の通信による遅延はプログラミング手法による遅延に影響しないと考えたためである。

(1) 1個のプロセッサ内、マルチスレッド処理
(2) 1個のプロセッサ内、共有メモリとシグナルを用いたマルチプロセス処理
(3) 1個のプロセッサ内、PVM を用いたマルチプロセス処理

Fig. 5 において、IP は IP5005BD を用いて 3 種類のカラー画像のしきい値処理を行い、その結果を、(1) の場合は共通変数に、(2) の場合は共有メモリ SHM に、(3) の場合は IP の私的変数に書き込むという処理を繰り返す。COG は IP の画像処理の終了をシグナルによって知って、認識結果 (実際には適当な値) を共有メモリなどに書き込み、次の画像処理終了のシグナルを待つ、という処理を繰り返す。PLAN は COG の認識結果更新をシグナルによって知って、計画および実行指示を行う (実際には適当な値を共有メモリなどに書き込む)、という処理を繰り返す。すなわち、IP が画像処理を反復し、その終了通知によって COG と PLAN が逐次的に動作しているという非常に単純なマルチプロセス環境である。

上記のプロセス構成で認識-計画-実行というループを 10000 回繰り返し、その平均時間を求めた結果、いずれの場合もほぼ 67ms となった。これは、COG および PLAN の処理時間が IP での処理時間 (約 66ms) に比べて非常に短いためである。また、平均時間が IP の処理時間にほぼ等しかったのは、画像処理の時間に比べてコンテクストスイッチなどのオーバヘッドが無視できる程短かったことを表している。今回の実験では各プロセスのメモリ所要量が小さいのでスワップアウトが起きなかったと考えられる。当然ながら、これが起きるような状況ではコンテクストスイッチによるオーバヘッドの影響は大きくなる。

このことから、今回の実験の環境では、マルチスレッドプログラミングとマルチプロセスプログラミングとで、遅延の問題に有意な差がないことが知られた。ただし、プロセス数が多くなった場合、および、各プロセスのメモリ使用量や計算量が大きくなったときの状況をこの実験結果のみから予測することはできないので、さらなる検討が必要である。

なお、(1) から (3) のいずれの場合でも反復の初回はプロセスの起動などのために 2 秒弱の時間を要した。マルチスレッドプログラミングの方が 1 割程度時間が短かったが、これは初回のみであるので、ロボット制御においては実質的な問題ではない。

5 まとめ

本報告では、著者らの研究室で製作した自律移動ロボットの構成を紹介し、PVM を用いた分散並列動作の可能な制御用プログラムの作成について述べた。PVM を用いたプログラムはマルチスレッドプログラミングによるものに比べると、コンピュータへの負荷が大きく、応答時間の遅れが懸念されたが、今回の実験では有意な差は出なかった。PVM を用いると、メッセージパッシングなどのオブジェクト指向的プログラミング手法を利用できるので、ソフトウェアの開発効率を高めることができる。したがって、自律移動ロボットの制御プログラムを作成するツールとして PVM は非常に有望であると考える。

この研究により、自律移動ロボットの協調動作を含め

た種々の研究のためのインフラを整備することができた。今後は、このロボットシステムを用いて、これらの研究を進めていく。

参考文献

[Geist, 1994] Geist, A. et al. : PVM: Parallel Virtual Machine, A Users' Guide and Tutorial for Networked Parallel Computing, The MIT Press, 1994. *ftp://ftp.netlib.org/pvm3/book/pvm-book.ps*

[Netlib, http] Netlib Repository at UTK and ORNL : http://www.netlib.org/, *in Japan : http://phase.etl.go.jp/mirrors/netlib/.*

[HPF, http] High Performance Fortran: http://www-unix.mcs.anl.gov/dbpp/web-tours/hpf.html

[HPCPP, http] High-Performance C++: http://www.extreme.indiana.edu/hpc++/

[OpenMP, http] OpenMP Simple, Portable, Scalable SMP Programming: http://www.openmp.org/

[MPI, http] Message Passing Interface Forum: http://www.mpi-forum.org/

[MPIIMP, http] MPI Implementation List: http://www.mpi.nd.edu/MPI2/

[LinuxThread, http] The LinuxThreads library: http://pauillac.inria.fr/ xleroy/linuxthreads/

[森, 2000] 森信人 他 : 全方向移動機構と全方位視覚を有する小型ロボットによるサッカー競技の実現 - チーム OMNI の戦略 -, 人工知能学会 第 6 回 SIG-Challenge 研究会講演論文集, pp.42-47, 2000.

KU-Boxes2001 における高速色抽出処理

Fast Color Image Segmentation in Team KU-Boxes2001

田中一基, 影山 茂, 五十嵐治一, 黒瀬能秀
Kazumoto Tanaka, Shigeru Kageyama, Harukazu Igarashi, Yoshinobu Kurose

近畿大学工学部（広島県東広島市）
School of Engineering, Kinki University, Higashi-Hiroshima

Abstract

It is important to process color image in real time for robot systems such as soccer robot systems for RoboCup. In this paper, we describe a method for fast color image segmentation used in our robot system. Furthermore, we propose a color extraction method where color-matching regions on HS-space are approximated by rectangle pieces on UV-space. Using the method, it is possible fast and robust color extraction without transforming UV to HS.

1 はじめに

自律移動型ロボットシステムの評価の場として，ロボットサッカーの競技会 RoboCup[1]があり，実機小型部門において，我々は共通プラットフォーム開発のプロジェクト JP/S-II を進めてきた[2]. しかし, これまでの評価の結果, 我々のシステムは, 性能面における幾つかの問題が明らかになっている[2]. 本稿では，その中でも，画像処理速度の問題を取り上げる.

本システム（グローバルビジョン方式）の画像処理は，ロボットに取付けられたカラーマークやカラーボール等の色抽出処理が主である[3]. この処理に対して，我々は，1コマのカラー画像あたり，33ms（ビデオレート）以下で処理することを目標としている. しかし，これまでの処理方式では，等色領域ごとに抽出処理を繰り返していたため，1コマのカラー画像の処理に，約 100ms の時間を要していた[4].

高速な色抽出処理の方式としては, これまでに, 色の属性値を添字とする配列データの各ビットを, 等色領域を示すフラグに見立てることによって, 複数の等色領域との照合を同時に行う方式が，カーネギーメロン大学のチームにより提案されている[5]. そこで，まずこの方式を，現行のロボットシステム KU-Boxes2001 の画像処理システムで用いている画像処理ボード（日立 IP5005）上で，実現することを試みた. 2で，この実装方法について，概略を述べる.

次に, 我々は前述の配列を2次元に拡張し, YUV 画像において，高速かつ頑健な色抽出を可能とする処理方式を考案した. KU-Boxes2001 の画像処理では，Y,U,V の閾値処理を用いて色抽出処理を行っているが，それよりも頑健な色抽出方法として, 色相,彩度,輝度に関する属性を用いる方法がある[6]. 照明環境の変動等の問題に対しては，これらの属性を利用することが望ましい. しかし，属性 U,V から, 色相,彩度に関する属性への変換処理が必要であるため，高速色抽出に対しては処理時間の点で問題がある. そこで，U,V を変換することなく，色相,彩度,輝度で設定される等色領域を，近似的に設定する方法を提案する. 3 で，この方式の詳細と，実験結果とを示す.

2 現行の色認識処理

KU-Boxes2001 における色認識処理について，処理手順(1)～(4)に従って概略を述べる.

(1) YUV 濃度変換

画像処理ボードに取り込んだ Y,U,V の各濃度を，8 種類の等色領域を表現可能な 8 ビットデータに変換する. 濃度変換には，Y,U,V の各閾値から設定される変換テーブルを用いる. この変換テーブルについて，以下に述べる.

等色領域を設定する閾値の範囲内の Y,U,V 値については，各色に対応するビットに 1 を置き，その他のビットには 0 を置く. 図 1 に変換テーブルの例を示す. 同図では，変換後の濃度データの下位ビットから順に，色 A～H を対応させている. たとえば，色 A の Y,U,V の各閾値が,

（Ya_min, Ya_max） = （1, 2）

(Ua_min, Ua_max) = (2, 4)
(Va_min, Va_max) = (3, 4)
の場合，同図で示す位置に 1 を置く．

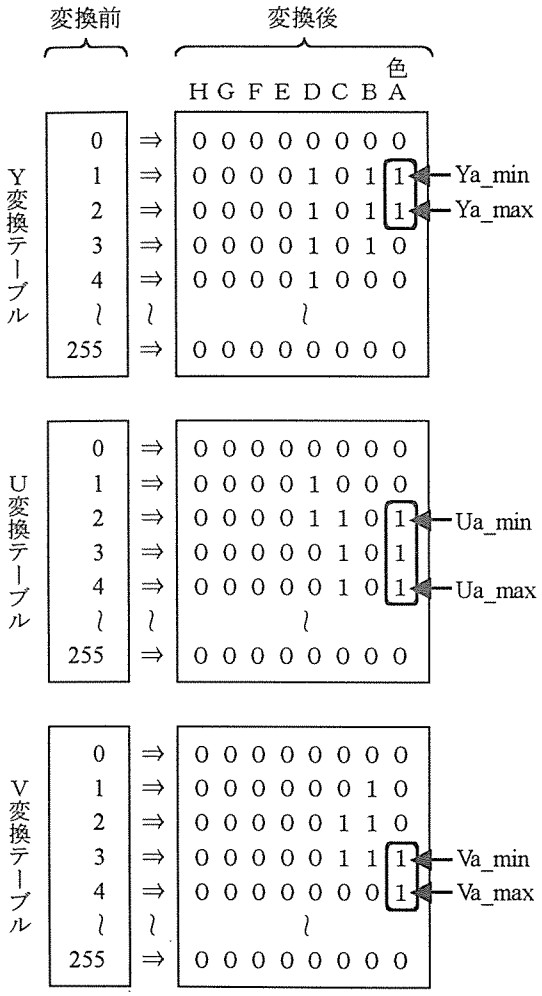


図 1　YUV 濃度変換テーブルの例

(2) 画像間 AND 処理

(1)で濃度変換された Y,U,V の各々に対し，画像間の AND 処理を行う．各画素において，この処理を行うことにより，等色領域を示す数値，あるいは 0 から成る画像を得る．1 画素あたりで必要な AND 処理は 2 回である．処理後の画像の例を図 2 に示す．同図において 1,2,4,8 の各値は，図 1 の変換テーブルの設定から，それぞれ，色 A,B,C,D を表している．

図 1 に示した濃度変換テーブルの例では，ある画素の濃度が Y=1,U=2,V=3 であった場合，それぞれ 11,13,7 に変換され，AND 処理によって 1，すなわち色 A を示す数値が得られる．

0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	1	1	0	0
0	1	1	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0	0
0	2	2	0	0	0	0	0	0	0
0	2	2	0	0	0	0	0	0	0
0	0	0	0	4	4	0	4	4	0
0	0	0	0	4	4	0	4	4	0
0	8	8	0	0	0	0	0	0	0
0	8	0	0	0	0	0	0	0	0

図 2　AND 処理後の画像の例

(3) 2 値画像処理

AND 処理後の画像を閾値 1 で 2 値化し，ラベリングおよび 2 値連結成分の属性値（重心座標等）の計算を行う．

(4) 色情報取得

連結成分の色情報を，AND 処理後の画像から得る．このため，(3)で得られた重心座標の各々を中心とした小ウィンドウ内（たとえば 3×3）の最大濃度値を抽出する．この処理は，対象物の像が凸集合であるため有効である．

以上の色認識処理は，IP5005 に用意されているライブラリ関数を，そのまま利用することができる．現在，カラー画像 1 コマ（Y : 512×512，U : 256×512，V : 256×512）あたり，ロボットの ID 認識等の後処理を含め，約 30ms で行うことが可能となった．したがって，KU-Boxes2001 では，ビデオレートによる画像処理が可能である．

3　2 次元配列要素の参照による色抽出方式

照明環境の変動等に対して頑健な色抽出のために，図 3 に示すような，色相,彩度,輝度に関する閾値を用いる方法が知られている．これらの閾値処理において，2 で述べた方法を適用するには，属性 UV を色相,彩度に関する属性に変換した上で，これらの濃度変換テーブルを利用する方法が考えられる．しかし，IP5005 の場合，このような属性変換には，U および V の 1 面（512×512）あたり約 9ms の処理時間が必要である[6]．したがって，2 で述べた色認識処理時間を加えると，1 コマあたり 39ms の処理時間が必要となり，ビデオレートで処理するには，処理時間の点で問題が生ずる．

そこで，属性 U,V を変換することなく，色相,彩度に関する閾値処理方式と同程度に頑健な抽出方式を考案した．この方式は，”U および V 値

と，等色領域をUV平面へ投影した領域（以後，UV等色領域）との，対応情報を持つ配列**UV**”と，”UV等色領域およびY値と，等色領域との，対応情報を持つ配列**IY**”との，2つの2次元配列を用いる．配列**UV**により，UV平面上の任意の座標を，UV等色領域として指定することができる．すなわち，**図4**に示すように，色相,彩度に関して設定される領域は，正方ブロックの集合で近似される．したがって，UV平面で，U,Vの最大/最小値による矩形領域や，色相,彩度の最大/最小値による扇形領域といった，比較的単純な等色領域の設定だけではなく，複雑な形状の領域設定も可能である．

また，配列要素の参照により，AND処理を行うことなく，複数の等色領域との同時照合が可能である．このため，**2**で述べた方式よりも高速な処理が期待できる．

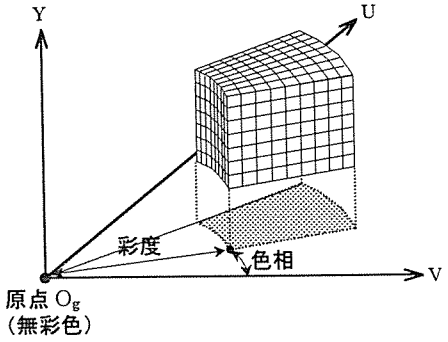


図3　色相,彩度,輝度で設定される等色領域の例

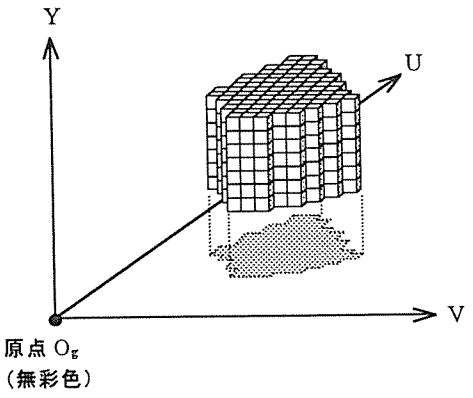


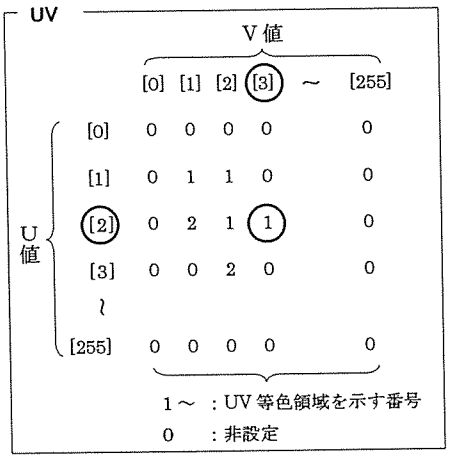
図4　本方式で設定可能な等色領域の例

3.1　本方式の詳細

まず，**図5**に例示するような，2種類の2次元配列**UV**，**IY**を考える．**UV**の添字に，U,V値をそれぞれ対応させる．UV等色領域の各々に番号を設定し，**UV**の対応する要素に，この番号を持たせる．ここで，Y値の違いにより，複数の等色領域が共通のUV等色領域を持つ場合には，共通領域に対して，新たに番号を設定する．

次に**IY**の添字に，UV等色領域の番号，Y値をそれぞれ対応させる．等色領域の各々に番号を設定し，**IY**の対応する要素に，この番号を持たせる．

これらの配列を用いれば，色抽出は，**IY**(**UV**(U値,V値)，Y値)を参照することで可能となる．たとえば，ある画素のY,U,Vの濃度値がそれぞれ1,2,3であった場合（**図5**），**IY**(**UV**(2,3),1)を参照することにより，対応する等色領域を示す番号1が得られる．



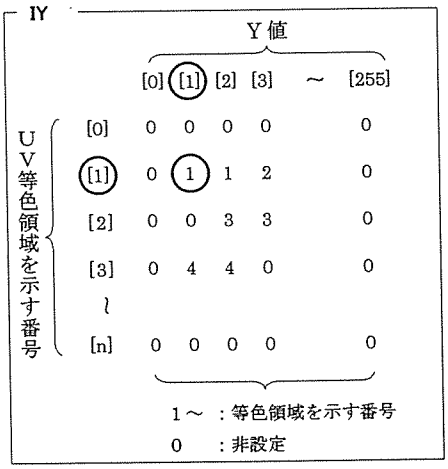


図5　配列要素参照による色抽出の例

3.2　色抽出の実験

本方式と従来のY,U,V閾値処理との，色抽出の

頑健さの比較を，カラーボール（青，直径34mm）と背景（青と白）の画像を用いて行った．

3.2.1 準備

まず，カラーボールのU,V値を計測し，これらを用いて計算した色相,彩度の最大/最小範囲（扇形）を近似する正方ブロックの集合領域を，カラーボールのUV等色領域とする．カラーボールのU,V計測値およびUV等色領域を図6に示す．なおIP5005では，U,V濃度（8ビット）の中間値が無彩色であり，図6ではこれを原点O_gとしている．また，照明変動に対応するため，Yの閾値は，計測値よりも広めに（±30%）設定した．

次に，比較対象とするY,U,V閾値処理で用いるU,Vの閾値は，従来と同様に，計測したU,V値の最大/最小値を用いた（図6）．また，Yの閾値は，本方式における閾値と同一とした．

次に，背景として用いた青色の紙のU,V値を計測した．計測値を図6に示す．

最後に，撮像は，図7に示すようなカラーボールと白および青の背景に対し，閾値設定時よりも照明を暗くした場合と，明るくした場合で行った．

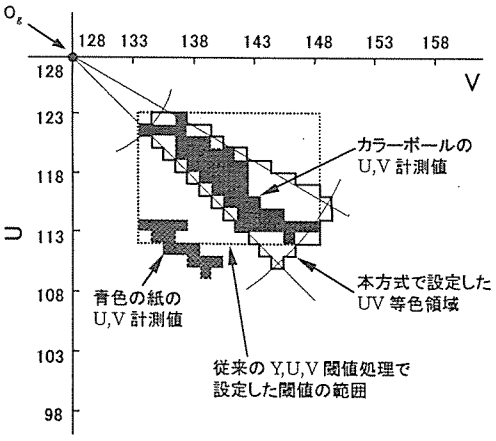


図6　実験対象の計測値とUV等色領域

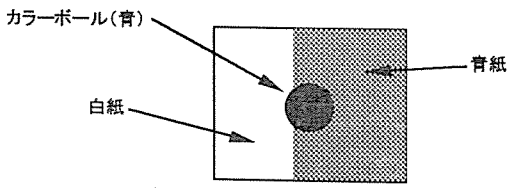


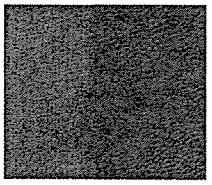
図7　実験対象の模式図

3.2.2 結果

閾値設定時よりも照明を暗くした場合と，明るくした場合の実験結果を，それぞれ図8，図9に示す．なお，抽出処理後の画像は，観察のため2値化している．

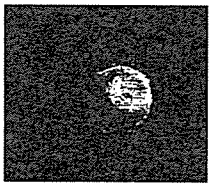
図8，図9のいずれも，従来のY,U,V閾値処理(b)では，カラーボールと背景が連結して抽出され，カラーボールの切り出しは困難である．これは，青紙の等色領域と，Y,U,V閾値処理で用いた閾値の範囲とが，一部，重なっているためと考えられる（図6）．これに対し，本方式による処理(c)では，いずれの照明変動においても，青紙の影響を抑え，比較的安定してカラーボールの切り出しが可能であることがわかる．

なお，本方式の処理を実行可能なライブラリ関数がIP5005には用意されていないため，IP5005で取得した画像をホストコンピュータ（PentiumⅢ 600MHz）のメモリに転送し，ホストCPUで本方式の処理を行った．Y,U,V値から等色領域番号への変換処理時間は，1コマのカラー画像あたり約5msであり（メモリへの転送時間は含まない），色抽出においては，十分に高速であると考えている．



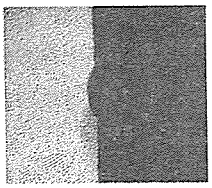
(a) 原画像

(b) 従来のY,U,V閾値処理

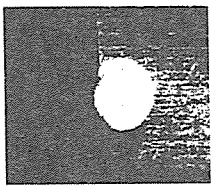


(c) 本方式による処理

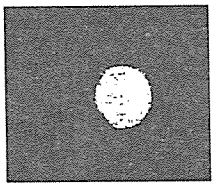
図8　閾値設定時よりも暗くした場合の抽出結果



(a) 原画像



(b) 従来の Y,U,V 閾値処理



(c) 本方式による処理

図 9　閾値設定時よりも明るくした場合の抽出結果

4　考察

3で提案した方式を，2 で述べた色認識処理手順に組み込むことを考える．2 の，(1)YUV 濃度変換と，(2)画像間 AND 処理とは，本方式にそのまま置き換えることができる．2 で述べたように，IP5005 では，(1)～(4)の処理を約 30ms で実行しているが，このうち，(1)と(2)との処理時間の合計は約 13ms である．したがって，配列要素の参照による本方式の処理が，約 13ms 以下の処理時間で実行できれば，現行の色認識処理方式と同様に，ビデオレートでの処理が可能である．実際，ホストコンピュータ上では約 5ms で処理できたので，IP5005 上でも十分に可能であると考えられる．

しかし，3.2.2 でも述べたように，提案した方式の処理を IP5005 上で容易に実装するためのライブラリ関数は，現在のところ用意されていない．たとえば，画像メモリに設定した配列 UV および IY の要素を，画像処理ボード上で直接参照し，その結果を別の画像メモリに直接書き込む処理を，高速に実行する関数である．

5　おわりに

本稿では，現行の KU-Boxes2001 に実装している色認識処理の概略を述べた後，頑健な抽出を行うための高速色抽出方式を提案した．本方式は，UV 等色領域あるいは等色領域を表現する配列要素の参照に基づいて，色抽出を行うことを特徴とする．本方式により，色相,彩度への変換を行うことなく，頑健な色抽出が可能となる．また実験により，U,V 計測値の最大/最小値を用いた従来の Y,U,V 閾値処理と比べ，照明変動に対する頑健性を確認した．また，ホストコンピュータ上で処理した場合，1 コマのカラー画像あたり約 5ms の処理時間で色抽出を行うことを確認できた．

今後は，本方式を画像処理ボード上へ実装する上での，最適な画像処理システムを検討した後，色抽出処理と認識処理を含めた処理時間の評価を行う予定である．

なお，本研究の一部は，日本学術振興会より科学研究費補助金（C2，課題番号 11680405）の助成を受けた．

参考文献

[1] 浅田稔，國吉康夫，野田五十樹，北野宏明：“研究活動とロボットコンテスト（RoboCup）”，日本ロボット学会誌，Vol.15，No.1，pp.13-16，1997

[2] 五十嵐治一，小末将吾，田中一基，黒瀬能聿，五百井清：“JP/S-II プロジェクトの総括と今後の展開”，人工知能学会研究会資料 SIG-HOT RoboCup Spring Camp 2000，pp.1-5，2000

[3] 田中一基，朝岡忠，小末将吾，五十嵐治一，黒瀬能聿：“RoboCup 小型部門用共通ロボットシステムにおける画像処理サーバ - JP/S-II プロジェクト - ”，人工知能学会研究会資料 SIG-Challenge-9804，pp.1-4，1999

[4] 影山茂，三吉孝則，飯土井修一，小末将吾，五十嵐治一：“KU-Boxes2000 における画像処理と旋回性能の改良”，人工知能学会研究会資料 SIG-Challenge ロボカップジャパンオープン-2000，pp.48-53，2000

[5] Bruce,J.，Balch,T.，Veloso,M.：“Fast and Inexpensive Color Image Segmentation for Interactive Robots”，*Proc.of* IROS '00，pp.2061-2066，2000

[6] 村松彰二，間瀬水紀子，持丸俊二，小林芳樹：“カラー画像処理における高速色抽出処理とその応用”，第 2 回動画像処理実利用研究報告会研究報告書，pp.16-19，1998

RoboCupシミュレーションリーグ人間参戦システム OZ-RP の提案

OZ-RP system : RoboCup Simulation League Human Interface to beat replicants

秋田純一， 西野順二， 久保長徳，下羅弘樹， 藤墳到
Junichi Akita， Junji Nishino， Takenori Kubo， Hiroki Shimora， Itaru Fujitsuka

OZ-RP Project
akita@fun.ac.jp, nishino@se.uec.ac.jp

abstract

This paper introduce the OZ-RP system architecture, which allows human players to join the RoboCup simulation league matches. Final goal of the RoboCup research project is to construct an artificial team that can beat a human soccer team in fair-play status. OZ-RP system is toward this goal in the simulation league. We show and discuss constraints and system configuration of regulation legal human interface systems in the league.

1 はじめに

本稿の目的は、ロボカップシミュレーションシステムにおいて人間１１人がリーグの制約に合致しつつ参加するためのモデルとして、OZ-RP システムを提案することである。

ロボカップシミュレーションリーグは、人間と対等にサッカーをすることのできることを目的にした人工知能とロボティックスの研究課題 RoboCup のうち、サッカーサーバシミュレーションの上で行なわれるものである。協調エージェントとしてのプレイヤをプログラムで作成し、シミュレーションとして対戦する。

シミュレーションリーグは他の実機リーグに比べて戦術、戦略面での高度化が進み、２０００年現在で上位チームの協調的行動のレベルは、人間チームが実際のフィールドで行なっているゲームに匹敵している。これはシミュレーションリーグがハードウェアトラブルなどを回避することができ、また、過去の資産が比較的容易に継承できることに起因している。

ロボカッププロジェクトの最終目的は、人間チームと人工チームとの対戦という安全性なども考慮しなければならないきびしい制約下で、人工チームが勝利できるための原理と技術の開発にある [3]。そのための基礎技術の育成のためのシミュレーションリーグの環境でも、人間と人工プレイヤとがサッカーを対戦することには大きな意義がある。

OZ-RP (OZ by Real Players) チーム [6] は、人間11人による実プレイヤチームである。このチームと人工プレイヤの対戦を行なう。このとき、人間のチームも人工プレイヤチームと全く同じ制約にのり、そのうえで勝つべく全力を持って戦うことが、人間の持つ協調能力を計測し、他の人工チームとの比較を行う上で重要である。

このチームで人間プレイヤとシミュレーションフィールドを繋ぐために OZ-RP システムを提案する。基本機能は、サッカーサーバからの情報を人間プレイヤ操縦者 (以下パイロットと呼ぶ) に提示し、パイロットの決定した行動をサッカーサーバで規定されたプロトコルに乗せ、送信するというものである。さらに、パイロットの柔軟な判断を活かし、可能な限り強いチームとなるようなユーザインタフェースの工夫が必要である。提示する情報を分かりやすくマップしなおして情報補完を行ない、状況に応じた

視点位置変更などの必要もある。またパイロットの戦略的な意図を入力として、それを低レベルのプロトコルまで自動的に変換するアシストシステムも必要となる。

本稿では、本プロジェクトとシステムの持つ意義と目的を述べ、システムに求められる制約を明らかにしたのち、システム構成の提案と実現化の検討を行う。

2 OZ-RP プロジェクト

OZ-RP (OZ by Real Players) チームは、11 人の人間が操縦するインタラクティブプレイヤによるチームである。OZ-RP プロジェクトとしてシステム開発とチーム編成を行っている。

2.1 プロジェクトの意義と目的

OZ-RP プロジェクトの目的は、人間 11 人が操作するプレイヤによってシミュレーションリーグに参戦し、他の人工チームと対戦することにある。

これは RoboCup が掲げている、人間チームとロボットチームの戦いでロボットが勝つ、という最終目標にむけての一つのアプローチである。これまで、シミュレーションリーグでは、人工チームだけが開発、実験されてきた。しかしながら人間と人工チームとの対戦を最終目標とするならば、シミュレーションリーグにおいても人間との対戦を行なうことが有意義であり、かつ必要である。

実機各部門と比較すると、現在のシミュレーションリーグの人工チームの開発では、基本的なスキルの実現はほぼ完成しており、チームとしての協調的行動の実現が主テーマとして扱われるようになっている。試合でのプレーも人間のチームのように高度な協調が発現しており、またその完成度がチームの強度と密接に結びついている。これらのチームと人間チームが戦うことで、人間の持つ柔軟な判断力およびプレイヤ間で協調するための先見的知識など、どれだけの知的能力を発揮できるかを調べることができる。

そこで人工プレイヤと同じ条件を各人間パイロットに対しても課し、上手に協調できるかどうか、その原理はなにかを探る。

このため、使いやすいユーザインタフェースと、人間の高度な判断を活かすための情報統合や行動生成などの、半自動プレイサポートシステムを構築することが目標である。

2.2 課題と方法

処理の面での課題は、プレイヤの視点に立ったときの情報補完と、人間の高度な戦略判断を活かす行動の生成である。国際大会などでは、他の人工チームにできるだけ近い制約を負うことが必要であり、これらの制約を負っていることを明示できるシステムとしなければならない。また大会では、フェアプレーの観点から他の人工チームと比較して、極端に多くのホストを要求したり、特別なホストを多数接続したりすることもできない。

複数人で１つの入力あるいは出力装置を共有する、つまり一つのディスプレイを二人以上でのぞき込むような形態では、他のプレイヤの視覚情報や行動などが間接的に得られることになり、人工プレイヤには実現不可能な情報共有となり、不公平であると同時に本来の主旨から離れてしまう。このため、１１人のプレイヤをプレイヤとしてサッカーサーバに接続するには、１１台の入出力装置を必要とする。

１１台の入出力装置を、より少ないホストを介して接続するため、アプリケーションレイヤでのマルチプレクサが必要であり、その構築が不可欠である。これはホスト１台に繋ぐための、ハードウェアとしてのマルチプレクサであり、なおかつ複数の入出力装置のデータを区別できるようなサーバシステムも必要である。

2.3 現状

プロジェクトは公募型で行なわれ、システムを作るエンジニアと、プレイを行なうパイロットとを合わせ 2001 年 3 月末現在で１５名が参加している。パイロットだけでの応募も認めた背景では、システム作製作業に因われず、プレイヤとして繰り返し練習をし、最大の能力を発揮してもらうことを期待している。メンバーは日本各地に分散しており、集まって密な相談のうえでシステム構築をすることはできない。

OZ-RP としてのプロトタイプシステム開発は4〜6名が主として行なっており、メイリングリストと専用の WWW ページを介して相互に情報交換を行なっている。パイロットや直接にプロトタイプを開発していないエンジニアも、各自のアイディアを提案し検討している。

3 OZ-RP システムの提案

3.1 概念設計としての OZ-RP システム

OZ-RP システムとは、1つの実体としてのシステムを指すものではなく、ロボカップシミュレーションの制約に合致し、人間の能力を引き出すためのユーザインタフェース構築の枠組である。

パイロットは、各自の好みを代表とする個々の特性を持っている。とくに直接の表示および入力のインターフェースでは、全員に一律で同一のシステムを提供するより、個々の好みに合わせて選択できるようにした方が、習熟度も上がりやすい。

このため、システムとして1つの実体は作らず、人間参入における制約を満たすための概念設計のみとした。

3.2 システムの目的

OZRP システムは、OZRP プロジェクトのためのユーザインタフェースの概念設計である。このため、分散的に開発されるサブシステム群や統合システム事例を、以下の目標のために構造化し示すことがシステムの目的である。

まず、パイロットが各個人の嗜好や特性に応じて入力、出力、アシストの各サブシステムの種類を選んで、自由に組み合わせられることを目指している。さらに、システム設計上の制約として、ハードウェア接続の複合化 (マルチプレキシング) が必要であり、これにも対応できるような構造を示すことも目的となっている。

3.3 システム設計上の制約

シミュレーションリーグのレギュレーションに従うことが、そのままシステム設計上の制約となる。項目としてまとめると以下のようになる。

1. プレイヤ間のオフライン情報交換の禁止
2. 可搬性とセットアップ所用時間の最小化
3. ホストの特権ユーザなど管理上の特別の措置の最小化
4. サッカーサーバプロトコル準拠による仮想プレイヤ化

項目1は、レギュレーション上重要な制約である。人工プレイヤの場合はサーバを経由しないプロセス間での通信が禁じられている。人間の場合は音声やジェスチャーを含め多様なモードでのコミュニケーションができてしまうため、システムとしてできるかぎりアンフェアな情報の共有・入手を排除する必要がある。たとえば、ディスプレイを共有することは、本来他のプレイヤの視点でしか見えない情報を目にすることができることになり好ましくない。このため、パイロット毎の情報表示スクリーンが必要となる。

項目2と3は、大会の運営上の制約である。さらに、ロボカップの趣旨である制約された計算能力での可用システムの実現からも、他の人工チームと同じホストのみで処理する必要がある。これにより、パイロット 11 人分の操作端末のために、11 台のホストを要求しないことを前提にシステム設計する必要がある。

項目4は、サーバ側での変更・調整などを一切必要としないという意味である。これにより他の人工チーム同様サーバでの試合ログの保存などが可能となる。

3.4 OZRP システムの概要

以上の目的と制約にもとづき、図1に示す複数のサブシステムによる構造を提案する。

このシステム構成により、サブシステムの連結部で多重化 (マルチプレキシング) や分散化できる。また、表示システムと入力システムのインタフェース規格をそれぞれ定義することで、パイロットの好みに応じたものを選ぶことが出来るよう配慮している。同様にアシストシステムもさまざまに交換可能とした。

とくに多重化できる部分を増やしたことで、使用する実デバイスやネットワーク構成によって、11 台

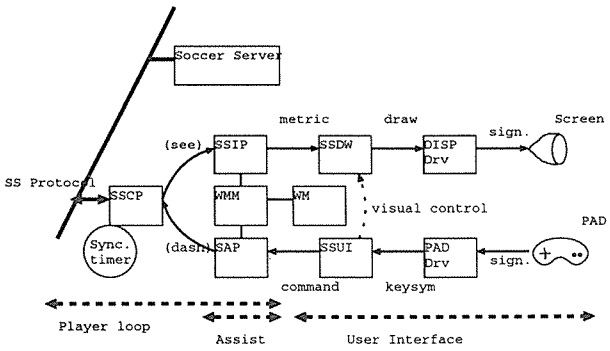


図 1: OZRP システム構成, SS: Soccer Server, SS Protocol: Soccer Server Protocol, SSCP: SS Comunication Processor, SSIP: SS Information Processor, SSDW: SS Display Window manager, WM: World Model, WMM: WM Manager, SAP: Soccer Assistance Processor, SSUI: SS User-command Interpreter, DISP Drv.: Screen Driver, PAD Drv.: key Pad Driver

分の多重化の実現方法も増加している。

3.5 SSCP

サッカーサーバに対しプレイヤとして直接接続する部分が、SSCP(サッカーサーバコミュニケーションプロセッサ) である。サッカーサーバプロトコル (SSP レイヤ) では一つの UDP 双方向ソケットで、情報の受信と行動命令送信を行なう。いっぽう OZRP システム内では、情報表示と行動操作が必然的に分離されているため、SSCP がこれらの分離、統合を行なう。さらに内部にはタイマを持ち、サーバの 100msec 刻みに同期して送受信制御を行なう。

3.6 プレイヤループ・アシストシステム

プレイヤループは、通常の人工プレイヤと同じ構造を持っている。すなわち、ノイズの含まれたサーバからの視覚などの情報統合、ワールドモデルの保持と更新、ボール追尾やドリブル、目的地への移動などの自動動作を行なう。パイロットにたいしては、統合した情報を提示し、パイロットからの戦略的なレベルでの行動命令を解釈し、なおかつ自動で動作を行なう。

最低限のシステムとしては、SEE メッセージで受け取った情報をそのまま提示し、DASH などの SSP レイヤでのコマンド発行を人間が行なうことも含まれている。しかしながら、直接発行では操作性が悪く人工プレイヤに匹敵するプレイは不可能である。

個々の人間の情報把握能力や行動速度は、人工プレイヤのそれらとは比較にならないほど遅い。いっぽうで人間は、十分な時間があれば限定された情報から協調のために必要な情報を取り出し、高度な戦略を立て動作決定する事ができる。人工プレイヤに匹敵した試合を行うためには、この人間の高度な判断能力の有効性を引き出すプレイヤループが必要である。

3.7 パッド

ユーザが触れる入力装置をパッドと呼ぶことにする。パッドの候補としては、キーボード、マウス、各種ゲーム機器パッドなどが挙げられる。

キーボードとマウスは各ホストに 1 式ずつ標準で装備されているものであり、単体プレイヤにとっては特別のハードを必要とせず理想的である。しかし、11 人分を用意するにはホストを 11 台占有する必要がある。

各種ゲームパッドは、一般に操作性に優れ使用感覚の評価も高い。ホストとの接続にはなんらかの特別なハードウェアが必要となる。このとき、接続ハードウェアにマルチプレクサ機能を同時に組み込めば、1 台のホストに対し複数のパッドを接続できるようになる。

3.8 スクリーン

パイロットへの情報提示出力機器をスクリーンと呼ぶことにする。候補として、コンピュータディスプレイ (CRT、液晶、プロジェクターを含む)、携帯ゲーム機の液晶ディスプレイが挙げられる。

コンピュータディスプレイは、通常ホスト 1 台につき 1 台の接続となるため、共有をしないという制約に従うとホストが 11 台必要となる。

液晶ディスプレイを持つ携帯ゲーム機では、11 台

用意できれば11面のスクリーンを準備したことになる。この場合、11台接続のためのマルチプレクサと、ホストとの物理的な接続に特別なハードウェアが必要となる。この接続ハードウェアにマルチプレクサ機能を同時に組み込めば、1台のホストだけで複数のスクリーンを接続できるようになる。

3.9 Visual control バイパス

図1のシステム中で、SSUIからSSDWへのバイパスを設けている。これは、カーソルポインティングや視点変更などを実現するときに、パッドからのスクリーンの制御を行うためのものである。このバイパスを通してユーザインタフェース部での独立したループを構成している。

カーソルポインティングなどは、アシストおよび情報統合操作とは無関係である。システムのパッドとスクリーンはプレイヤループ中のワールドモデルを介して情報伝達できるように作られているが、つねにこのループを介することはシステムの純粋な負担となるため、バイパスを設けて効率を向上している。

3.10 ロギング

試合のログはサーバが通常どおり保存できる。人間の判断や行動の分析に使う事を念頭にOZ-RPシステムでは操作系列も保存する。操作系列はSAPアシストサブシステムで行なう事で、パイロットに提示されたワールドモデル情報と、入力されたパッドの時系列情報を組として保存することができる。

4 OZ-RPシステムの開発

OZ-RPシステムのプロトタイプは、分散的に開発されている。それらの成果は互いにWWWやメイルリストを通じて情報交換されている。現在これらのプロトタイプには、11人の人間パイロットがホストに接続することが出来ないものも含まれている。11人接続を念頭に課題の洗い出しを行なう事がプロトタイプ開発の目標である。

4.1 OZRPC

OZRPC (OZ-RP Controller) [1] は、家庭用ゲーム専用機の入力機器(通称コントローラ)をシリアル経由で接続するシステムであり、はこだて未来大学の秋田氏によって開発されている。

11台のコントローラを統合し、ホストコンピュータの1つのシリアル端子に接続できるように工夫している。このため専用のシリアルレベルインテリジェントマルチプレクサをPICマイコンを用いて実現している。

4.2 JINN

JINNシステム[2]はゲーム機のコントローラ接続システムと、通常のサッカープレイヤをベースとしたプレイヤループに、プレイヤビューを表示するサッカーモニタを統合した、単体プレイヤ操作システムである。豊橋技術科学大学の藤墳氏と渡内氏によって開発がすすめられている。

ディスプレイと接続ホストを1プレイヤにつき1セットずつ必要とする。

4.3 OZip

OZIP (OZ Interactive Player) [7] は高機能サッカーモニタ (SoccerViewer) と、独立キーパエージェント Savior [5] によるプレイヤループをベースにした gtk+ (X11) による、統合型の単体プレイヤ操作システムである。福井大学の下羅氏によって開発が進められている。

情報統合補完とボールキャッチアップ、目標位置移動がアシストシステムとして実装され、マウスで動作命令を与える。

ディスプレイと接続ホストを1プレイヤにつき1セットずつ必要とする。

4.4 OZRPWWS

OZRPWWS (OZRP Wonder Witch System) [4] は、液晶ディスプレイを搭載した携帯ゲーム機を入出力両用の装置とするインタフェースシステムである。

システム構築上の制約である11台のディスプレイ

の準備が容易であると期待される。

ホストとシリアル通信によって、操作入力と情報表示出力を交換する。１１台を１台のホストと接続するため、専用のシリアルマルチプレクサを開発中である。

4.5 検討中のシステムと方式

11 台の OZRP 端末を素早く簡単に、システムに負担をかけずに接続する方法は、現在のところ無いと言える。

問題提起のため、可能な手法とその得失をここにまとめておく。

- 通信接続形態: シリアル、パラレル、USB、PS/2 マウス、イーサネット
  USB は、ハブを介した１１台のマルチドロップ接続が可能であるが、ロボカップシミュレーションで使われる UNIX サーバで利用できるかどうか不確実である。シリアルはハードウェアとして一般的であり、もっとも可用性が高いが、専用のアプリケーションレベルマルチプレクサを準備する必要がある。
- スクリーン (表示) 形態 : 各種ディスプレイ、携帯ゲーム機、HMD
  互いに干渉しない情報提示としては、HMD(ヘッドマウントディスプレイ) が最適であるが、可用性やコストおよび接続のために専用のハードウェアが必要であるといった問題がある。
- パッド (入力) 形態 : ゲームコントローラ、キーボード、マウス、プロッタ、モーションキャプチャ
  モーションキャプチャによる入力装置は操作感覚としては最適である。しかしながら、コストや大会会場での準備の面では利用は不可能に近い。

5 まとめ

OZRP システムとして人間がロボカップシミュレーションゲームに参戦するためのフレームワークを提案した。

このフレームワークの設計とプロトタイプの実装を通して、ロボカップの目標である人間対人工システムのフェアな試合の実現のためには、さまざまな制約が存在することを示した。また、これらの問題をクリアするための幾つかの方針を示した。

２００１年までのシミュレーションリーグの上位チームは、すでに洗練されたチームプレーを実現している。人間のもつ柔軟なチームプレーを引き出すためのシステム上の工夫が必要である。

人工システムと同等に戦えるようになった後、人間の協調行動のプレーログと操作ログをあわせて分析することで、各種の知的システムの構築技術へフィードバックすることが期待される。

参考文献

[1] Junichi Akita. http://www.fun.ac.jp /~akita /robocup /ozrpc.html.

[2] Itaru Fujitsuka. http:// www. ita. tutkie. tut. ac. jp /~itaru /robocup/controller.html.

[3] H. Kitano, editor. *RoboCup-97: Robot Soccer World Cup I.* Springer, 1998. ISBN 3-540-64473-3.

[4] Takenori Kubo. http://rook.fuis. fukui-u. ac. jp /~nishino /ozrp /ozrpws.html.

[5] J. Nishino, T. Kawarabayashi, T. Morishita, et al. *RoboCup-99: Robot Soccer World Cup III*, chapter Zeng99: RoboCup Simulation Team with Hierarchical Fuzzy Intelligent Control and Cooperative Development, pages 649–652. Springer, 2000.

[6] Junji Nishino. http://bishop. fuis. fukui-u. ac. jp /~nishino /ozrp/.

[7] Hiroki Shimora. http://bishop.fuis. fukui-u. ac. jp /~shimora /robocup/.

情報量による移動ロボットの注視制御のためのセンサ空間構成

Sensor Space Segmentation for Visual Attention Control of a Mobile Robot based on Information Criterion

光永 法明　浅田 稔

Noriaki MITSUNAGA　Minoru ASADA

大阪大学大学院工学研究科

Graduate School of Engineering, Osaka University

{mitchy, asada}@er.ams.eng.osaka-u.ac.jp

Abstract

Since the vision sensors bring a huge amount of data, visual attention is one of the most important issues for a mobile robot to accomplish a given task in complicated environments. This paper proposes a method of sensor space segmentation for visual attention control that enables efficient observation taking the time needed for observation into account. The efficiency is considered from a viewpoint of not geometrical reconstruction but unique action selection based on information criterion regardless of localization uncertainty. The method is applied to four legged robot that tries to shoot a ball into the goal.

1　はじめに

移動ロボットには視覚センサが搭載されることが多いが，視覚センサを効率よく使うには適切な注視選択が重要となる．我々は，自己位置の同定ではなく行動決定のための，効率的な観測を行う行動決定法を提案している[1]．この手法ではセンサ値が前もって離散化されており、注視や観測かかる時間は一定であると仮定していた。しかし，さらに効率的な観測を行うには，注視制御のための自律的センサ空間分割が必要となる．

強化学習の分野では、学習時間が状態空間の大きさに対して指数関数的に大きくなる[2]ことから、自律的センサ空間の分割による状態空間構成法が提案されている[3] [4] [5] [6]．しかし，観測にかかる時間や，能動的視覚システムは考慮されていない．Kamiharako et al. [5]は、注視の粗密制御が結果として得られることを示しているが、用いたセンサは周囲の観測に能動制御を必要としない全方位視覚センサであった。

そこで我々は，注視制御のためのセンサ空間の分割と行動決定木の生成法を提案する．観測にかかる時間を考慮した，自己位置の同定のための観測ではなく，行動決定のための効率的な観測を実現する．

2　情報量に基づく効率的観測と行動決定

ロボットや環境，与えられるデータ等に関して以下のように設定する．1) ロボットは視覚センサを持つが視野角が限られており，受動的な観測のみでは行動決定に必要な情報が得られるとは限らない．2) 環境中にランドマークが配置されており，視覚センサの観測方向を変えることにより、視野を拡大し行動決定に十分な情報が得られる．観測方向の候補は挙げられている。3) 教示などにより，視野を拡大した際に得られる視覚センサの値と，その際にとるべき行動および，その行動後の拡大視野での視覚センサの値 (これら 3 つの組を 1 つのトレーニングデータとする) が与えられる．また行動決定に必要な情報は一定ではなく，状況に応じて変化する．

情報量によりコンパクトな分類木を生成する手法として ID3, C4.5 [7] がある。分類木の生成には，トレーニングデータセットが必要であり，各データは分類先のクラスと分類のための属性の値からなる．分類木が行動決定木の場合にはクラスは行動に，属性はセンサに相当する．ID3 の場合には属性値として離散値のみを扱う．センサ i について知ったときの行動に関する情報量 I_i をすべてのセンサについて計算し，最も情報量の大きいセンサの値によってトレーニングデータセットを分割し，全てのセンサについて情報量が 0 になる (データセット中の行動が 1 種類となる) まで，データセットの分割を繰り返す．行動決定木は，ノードがデータセットを分割するセンサ，枝がセンサの値によって伸びた形となり，葉で行動を表す．C4.5 では離散値属性については ID3 と同じであるが、連続値をとる属性の場合には，センサ値と閾値の大小関係でデータセットを 2 分割した際に，情報量が最も大きくなる閾

値を求め，閾値でデータセットを2分割することで，連続な属性値を扱う．

本研究では，C4.5 を元にロボットのセンサの特性を考慮した拡張を行った行動決定木生成手法を用いる．視野の限られた視覚センサを用いて、ランドマークの方位と閾値の大小関係を知るためには、ランドマークが観測されている場合を除き、視覚センサの方向を変え観測する必要がある。これに対し、ランドマークがある範囲の方位に観測されるか否かを知るためには、その方向の一度の観測で済む。そこで、データセットの分割をランドマークがある範囲に観測されるか否かで行い、分割後の情報量を比較する。

2.1 観測による情報量

行動の種類を r，トレーニングデータの数を n とする．トレーニングデータ中の行動 j をとった回数を n_j とすると，各行動 $j=1,...,r$ の生起確率 p_j は，$p_j = n_j/n$ である．このときの p のエントロピー H_0 は，次のようになる．

$$H_0 = -\sum_{j=1}^{r} p_j \log_2 p_j \tag{1}$$

ランドマーク i が $[\theta_{Lk}, \theta_{Uk})$ の範囲に観測されるか否かが分かったときの事後生起確率を求める．ランドマーク i が $[\theta_{Lk}, \theta_{Uk})$ の範囲に観測された回数を n^I_{ik}，観測されたときに行動 j をとった回数を n^I_{ikj}, $n^I_{ik} = \sum_j^r n^I_{ikj}$ とすると，範囲内に観測される場合の事後生起確率は，$p^I_{ikj} = n^I_{ikj}/n^I_{ik}$ となる．同様に，観測されなかった場合の行動 j をとった回数を n^O_{ikj}, $n^O_{ik} = \sum_j^r n^{ikj}$ とし，観測されない場合の事後生起確率 p^O_{ikj} を求める．$n_{ik} = n^I_{ik} + n^O_{ik}$ として，このときのエントロピーを計算すると，

$$H_{ik} = -\sum_{x=\{I,O\}} \frac{n^x_{ik}}{n_{ik}} \sum_{j=1}^{r} (p^x_{ikj} \log_2 p^x_{ikj}) \tag{2}$$

となり，それぞれの観測による情報量は，$I_{ik} = H_0 - H_{ik}$ である．情報量が大きいランドマーク i, 観測範囲 k ほど、行動に関する曖昧さが減少する。観測範囲の上下限 θ_{Lk}, θ_{Uk} は，各ランドマーク i についてトレーニングデータ中に含まれる方位の中点とする．

2.2 観測時間の考慮

観測に要する時間が観測対象 (ランドマークとその観測される範囲) によらず一定の場合には，情報量を最も大きい観測対象により，行動決定木を生成する．この決定木はコンパクトかつ，木のノードの観測を繰り返すことで最短観測時間で行動を決定できる．しかし，観測時間が観測対象により異なる場合には，最短観測時間となるとは限らない．また観測対象決定時に、得られる情報量と観測時間のトレードオフを計算するのは、観測時の計算コストが高く，決定木を使うメリットが減少する．

そこで決定木の生成の際に用いる指標を観測時間を考慮したものとする．前の観測対象の次にその観測対象を観測する場合にかかる時間を T とし，単位時間辺りに得られる情報量 i_{ik} を次のように求める．

$$i_{ik} = \frac{I_{ik}}{T+a} \tag{3}$$

ここで a は 0 で割らないための正の定数である．直前の観測対象がない場合には，行動決定時の平均的な状態からの観測時間を T として用いる．すでに視覚センサが観測した方向である場合には $T=0$ とする。

2.3 行動決定木の生成

i_{ik} を最も大きくする i,k の組により，トレーニングデータを，ランドマーク i が $[\theta_{Lk}, \theta_{Uk})$ の範囲に観測された場合と，観測されなかった場合に分け，行動が決定できるまで分割を繰り返す．この分割が木の枝分かれとなる．分割を繰り返しても，行動が確定しない場合には，各行動の頻度確率を記しておく．

たとえば、Table 1 のトレーニングデータが与えられたとする。表中の数字は各ランドマークが観測された方向である。視野が限られており視覚センサによる観測可能な方向が $[0,15)$, $[15,30)$, $[30,45)$ の 3 つであり、視覚センサは観測開始時に $[15,30)$ を向いており、$a=1$ 観測方向を変えるのに 1 時刻必要であるものとする。まず、H_0 を計算すると、$p_x = 2/4$, $p_y = 1/4$, $p_z = 1/4$ から、$H_0 = 1.5$ となる。Landmark A, B がある範囲に観測されるか否かがわかったときの、情報量 I_{ik} と単位時間辺りに得られる情報量 i_{ik} を計算すると Table 2 となる。そこで、最も単位時間辺りに得られる情報量が大きい、Landmark A が $[27,30)$ に観測されるか否かを確かめる。観測された場合の、トレーニングデータはデータ番号 3 のみで行動は y が決定できる。観測されない場合には、トレーニングデータ 1,2,4 が含まれ、行動は決定できない。このトレーニングデータのサブセットで、Landmark A, B がある範囲に観測されるか否かがわかったときの単位時間辺りの情報量は、Landmark B が $[0,15)$ に観測されるか否かが分かったときと、Landmark A が $[30,40)$ に観測されるか否かが分かったときが最も大きく、共に 0.05 である。左側を優先して観測することにすると行動決定木は、Fig.1 となる。

Table 1: Example training data

Data #	Landmark A	Landmark B	action
1	5	5	x
2	25	15	x
3	30	10	y
4	40	30	z

Table 2: Calculated information and information per time of the example training data (Lm means landmark)

Observation	Info. I_{ik}	Info. per time i_{ik}
$0 \leq (LmA) < 15$	.31	.15
$15 \leq (LmA) < 27$	.31	.31
$15 \leq (LmA) < 30$	.50	.50
$27 \leq (LmA) < 30$	1.4	1.4
$30 \leq (LmA) < 45$	1.4	.70
$0 \leq (LmB) < 7$	.31	.15
$0 \leq (LmB) < 12$	.5	.25
$0 \leq (LmB) < 15$	1.4	.70
$7 \leq (LmB) < 12$	1.4	.70
$7 \leq (LmB) < 15$	.5	.25
$30 \leq (LmB) < 40$	1.4	.70

```
Observe (15, 30], if 27<=(Landmark A)<30
then
        take action y
else
        Observe (0, 15] and if 0<=(Landmark B)<15
        then
                take action x
        else
                take action z
```

Figure 1: Action decision tree of the example data

2.4 観測予測モデルの生成

観測を効率化するためには観測予測を行う．観測予測に用いるモデルは問わないが、一時刻前でのランドマークの方位 $\boldsymbol{x}(t-1)$ の多項式で，行動 a をとった場合の，次時刻のランドマークの方位 $\boldsymbol{x}_{|a}(t)$ が表せるとすると，

$$\boldsymbol{x}_{|a}(t) = \sum_i A_{ai}\boldsymbol{x}^i(t) + \boldsymbol{b}_a \qquad (4)$$

ここで，$A_{ai}, \boldsymbol{b}_a$ はロボットを環境中で教示した際に観測したランドマークの方位から最小自乗推定を用いて求めた．$A_{ai}, \boldsymbol{b}_a$ には推定誤差が含まれるため，$\boldsymbol{x}(t)$ は確率分布 (一様分布) として表現する．

2.5 行動決定

行動決定は次のように行う。まず，一時刻前の $\boldsymbol{x}(t-1)$ の確率分布から，予測モデルを用い現時刻での $\boldsymbol{x}(t)$ の確率分布を計算する．次に，現時刻で視覚センサが向いている方向について，確率分布を修正する．この確率分布を用いて行動決定木の各葉への到達確率を計算する．同じ行動を示す複数の葉への到達確率の和を，その行動をとるべき確率とする．特定の行動が閾値を越えていれば，その行動をとる．そうでなければ，木の根に近い観測範囲を含むまだ観測していない方向から順に観測を行い，確率分布を更新する．更新した確率分布により，特定の行動をとるべき確率が閾値を越えるまで，観測と確率分布の更新を繰り返す．

3 実験

ロボットとしては，RoboCup SONY 脚式ロボットリーグのロボット (図 2) を用いた．カメラの画角は横 53 度，縦 41 度，画素数はそれぞれ 88, 59 である．脚は各 3 自由度，首は 3 自由度 (パン，チルト，ロール) ある．ランドマークを観測する際には脚，首のロールを固定し，パン、チルト軸のみを利用した．パン軸はロボット正面に対して，-88 度から 88 度、チルト軸は、-80 度から 43 度が可動範囲である．そこで能動的に観測する方向としては，パン軸を 44 度毎の 5 方向，チルト軸を 40 度毎の 4 方向の 20 方向にわけた．パン軸の最大角速度 5.9$[rad/s]$ は、チルト軸の最大角速度は 3.9$[rad/s]$ である。また軸が目標値に達した後で画像が安定するまで 0.16$[ms]$ 待ってから処理をしている。カメラの方向を変えて観測する場合に少なくとも 0.29[s] (パン軸の 44 度の回転に相当) かかることから、$a = 0.29[s]$ とした．

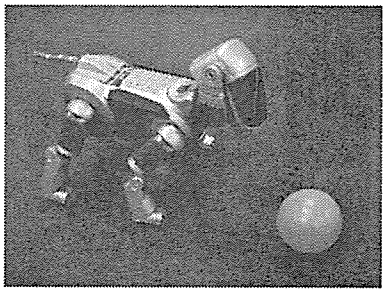


Figure 2: The SONY legged robot for RoboCup SONY legged robot league.

実験環境を Fig.3 に示す．RoboCup SONY 脚式ロボットリーグのフィールドである．ランドマークは 6 、ゴールが 2 あり，ボールが一つある．それぞれ，敵ゴール (TG), 自陣ゴール (OG), 北西ポール (NW), 北東ポール (NE), 中央西ポール (CW), 中央東ポール (CE), 南西ポール (SW), 南東ポール (SE) とする．すべてのランドマークとボールは色により識別される．ロボットがボールを TG に入れることをタスクとする．これを実現するためには場所に応じたボールへの回り込み，ボールの探索などが必要となる．

視覚センサとしては，各ランドマークとボールの図心座標，両ゴール (TG, OG) の画像上での x/y 座標が最小/最大となる座標 4 つを用いた．座標としては画像上での値を直接を用いるのではなく、対象が画像中心に観測されるときのパン，チルト軸の角度を用いた。またパン軸

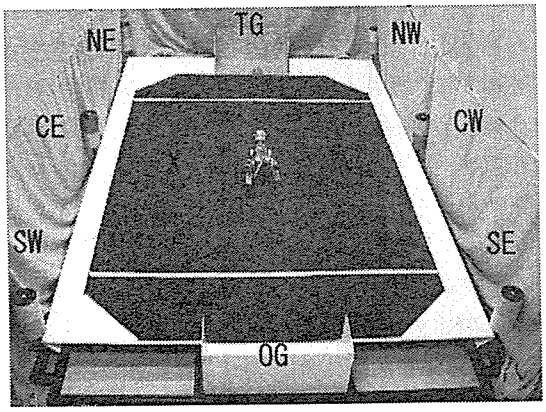


Figure 3: Experimental field (same as the one for RoboCup SONY legged robot league).

(x) 座標あるいはチルト軸 (y) 座標のいずれかのみの範囲ではなく，x,y の範囲の組 (長方形領域) に入るか否かでトレーニングデータを分割するものとした．これは，例えば x 座標の範囲が -10 度から 10 度にあるか否かで分割すると，y 座標に関しては指定されていないので y 軸についてカメラを回転して何度か観測を繰り返す必要があるためである．

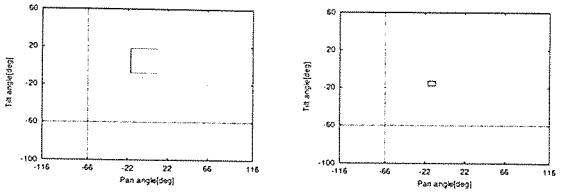
4 実験結果

フィールド中央 3 点のいずれかから行動を開始して，ゴール正面のボールをシュートすることをタスクとした．行動としては、ボール接近、前進、左 (右) 前進回転、左 (右) その場回転、左 (右) 横移動回転の 7 つを用意した。各点からシュートするまでの行動を 5 回ずつ教示した．行動ステップ数は 99 回となった．提案手法により得られた行動決定木の一部を Fig.4 に示す．図中数字の単位は角度 (degree) である．観測予測モデルを用いない場合には、一つ目の attention window は、$(-19,-8)-(-8,18)$ で、その中にボールの図心が観測されるか否かである (Fig.5(a))。このときの観測方向は (3,2) となる。パンの方向 X は左から 1,..,5 で 3 が中央，チルト Y の方向は下から 1,2,3,4 で 3 が正面とし観測方向 (X,Y) で表す．行動決定開始時にはカメラは正面中央 (パン:3, チルト:2) を向いているものとしている。正面中央 (パン:3, チルト:2) に観測されるボールやランドマークが判断の基準として優先的に利用されている。ボールの図心がこの window に観測されたなら、次の window は、$(-19,-10)-(-18,-13)$ となり、TG の y が最小となる座標が観測対象である。観測方向は同じ (3,2) である (Fig.5(b))。対象が window 何に観測された場合にはボール接近の、されない場合には前進行動をとる。ボール図心が window 内に観測されない場合には、次の window は、$(-19,-18)-(18,18)$ となり、TG の x が最小となる座標が観測対象となる。観測方向は (3,2) である (Fig.5(c))。観測対象が window 内に観測された場合には、次の window $(-19,-10)-(4,4)$ (Fig.5(d)) を観測する... と行動が決定されるまで続ける。

```
[ball] -19<x<11, -8<y<18 (Fig.5(a)) then
  [TG ymin] -19<x<-10, -18<y<-13 (Fig.5(b)) then
    Do TryReachBall
  else
    Do Forward
  [TG xmin] -19<x<18, -18<y<18 (Fig.5(c)) then
    [TG xmin] -19<x<4, -10<y<4 (Fig.5(d)) then
      Do Forward
    else
      [TG xmin] -19<x<11, -15<y<-3 then
        Do RightTurn
      [TG xmin] -6<x<18, 15<y<18 then
        [TG xmin] -12<x<11, -1<y<18 then
          [ball] 27<x<46, 26<y<32 then
            Do RightFoward
          else
            Do RightTurn
        else
          Do LeftRotate
          [TG xmin] 15<x<18, -8<y<4 then
            [CW] -52<x<-30, -15<y<5 then
              Do Forward
            else
              Do RightTurn
          else
            Do Forward
                    :
```

Figure 4: Part of action decision tree generated by proposed method

予めセンサ値を 20 度毎に離散化しておいた場合 (pre-quantization) と、提案手法による離散化のみを行った場合 (segmentation only)、提案手法による離散化と観測時間の考慮を行った場合 (proposed) の比較を行った。Fig.6-Fig.8 に、それぞれの手法により得られた attention window を示す。予め離散化した場合 (Fig.6) と提案手法による離散化 (Fig.7) を見ると、適切な離散化は均等に分けることではないことが分かる。さらに、観測時間を考慮する (Fig.8) と、情報量の大きい window ではなく、すでに観測した方向にある window から行動を決定することにより、観測時間を減少しようとすることが分かる。Table3 に、ノードの数 (# of nodes)、木の最大深さ (depth)、葉の数 (# of leaves)、観測方向数の期待値 (dirs)、観測時間の期待値 (time)、の比較を示す。観測方向数と観測時間の期待値は観測予測を行わない場合のものである。提案手法による



(a) Window for the ball

(b) Window for min. y of TG

(c) Window for min. x of TG

(d) Another window for min. x of TG

Figure 5: Attention windows

センサ値の離散化が木の大きさ、観測方向数、観測時間を小さくすることに有効であることが分かる。また観測時間を考慮することにより、木は少し大きくなるが、観測時間は半分以下に改善されている。

Table4 に、3 つの手法の実機実験による観測時間をの違いを示す。期待されたよりは長くなっているが、提案手法による観測時間は他の手法の半分以下となっている。観測予測を使うことでさらに観測時間が短縮されると期待される。

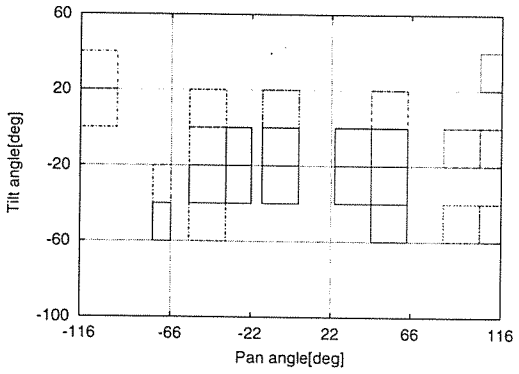


Figure 6: Created attention windows by pre-quantization method without time consideration.

5 まとめ

自己位置の同定ではなく行動決定のための、視覚センサと観測時間を考慮し、効率的な観測を行う行動決定法を提案した。実験により提案手法が、適切な離散化による行動決定木の圧縮と、観測時間の短縮に有効であることを

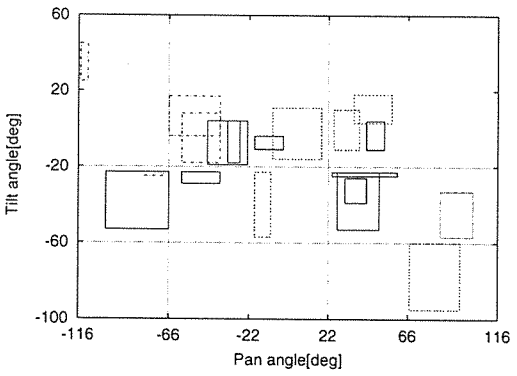


Figure 7: Created attention windows by proposed segmentation without time consideration.

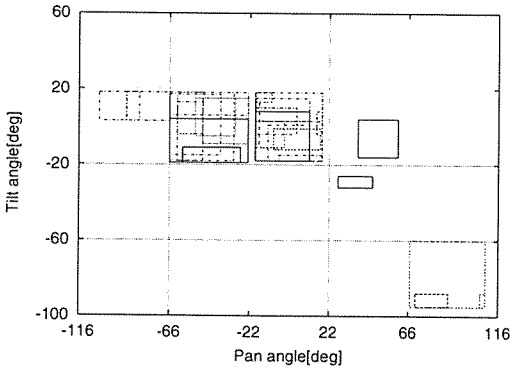


Figure 8: Created attention windows by proposed method.

Table 3: Comparison of sizes of the tree, expected number of observing directions, and expected time for observation

	# of nodes	depth	# of leaves	dirs	time[s]
pre-quant.	61	18	31	7.6	2.8
segment. only	39	11	20	5.5	2.0
proposed	59	15	30	3.4	0.83

Table 4: Comparison of mean time for obsrevation

pre-quant.	quant. only	proposed
6.6[s]	5.2[s]	2.5[s]

確認した。

参考文献

[1] 光永法明, 浅田稔. 移動体の意思決定のための情報量基準による観測戦略. 第 5 回ロボティクスシンポジア予稿集, pp. 351–356. 2000.

[2] S. D. Whitehead. A complexity analysis of cooperative mechanisms in reinforcement learning. In *Proceedings of AAAI-91*, pp. 607–613, 1991.

[3] Yasutake Takahashi, Minoru Asada, and Koh Hosoda. Reasonable performance in less learning time by real robot based on incremental state space segmentation. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1518–1524, 1996.

[4] Takehisa Yairi, Shinichi Nakasuka, and Koichi Hori. State abstraction from heterogeneous and redundant sensor information. In Y. Kakazu, M. Wada, and T. Sato, editors, *In Proc. of the Intelligent Autonomous Systems 5*, pp. 234–241, 1998.

[5] Masatoshi KAMIHARAKO, Hiroshi ISHIGURO, and Toru ISHIDA. Attention control for state space construction. In Y. Kakazu, M. Wada, and T. Sato, editors, *In Proc. of the Intelligent Autonomous Systems 5*, pp. 258–265, 1998.

[6] Minoru Asada, Shoichi Noda, and Koh Hosoda. Action based sensor space segmentation for soccer robot learning. *Applied Artificial Intelligence*, Vol. 12, No. 2-3, pp. 149–164, 1998.

[7] J. Ross Quinlan. *C4.5: PROGRAMS FOR MACHINE LEARNING*. Morgan Kaufmann Publishers, 1993.

KU-Boxes2001 における走行加速度の向上と経路計画について

Improvements in Moving Acceleration and Path Planning of Robots
in Team KU-Boxes 2001

早津啓道, 飯土井修一, 五十嵐治一, 黒瀬能秉
Hiromichi Hayatsu, Shuichi Iidoi, Harukazu Igarashi, Yoshinobu Kurose

近畿大学工学部（広島県東広島市）
School of Engineering, Kinki University, Higashi-Hiroshima

Abstract

This paper describes improvements to our robot system developed under the JP/S-II Project. We changed a constant-voltage system of a motor controlling circuit to a constant-current system. That change provided our robot with accelerating ability about two to three times larger than before. Moreover, we proposed a roadmap method for planning a path of a robot. We took account of time necessary for a robot to turn its body.

1 はじめに

近年, 人工知能とロボット工学の分野において, 動的環境下でのマルチエージェントシステムの標準問題としてロボットによるサッカーゲームが取り上げられており, RoboCup(The Robot World Cup Soccer Games and Conferences)という国際競技会が1997年から, 毎年, 開催されている. 我々は, 1997年9月から実機小型部門用の共通プラットホーム作成を目的とする JP/S-Ⅱプロジェクトを立ち上げている[1-5]. このプロジェクトについては, すでに, 2000年3月に東京で開催された RoboCup Spring Camp Symposium 2000 において, 一応の総括を行った[6].

その後, 本プロジェクトで開発したロボットシステムの性能評価のために, 実際に, 2000年6月に函館で開催された国内公式大会, RoboCup Japan Open 2000 実機小型部門に, KU-Boxes2000 というチーム名で参加し, 実際にサッカーの試合を行わせた[7]. そこでの試合の様子を観察して, ①走行時の加速度・速度が不足している, ②ロボットモータの消費電力が大きい, ③障害物回避と経路計画が不十分である, ④グローバルビジョンの画像処理速度が不十分, ⑤協調プレーの欠如, などの問題点があることがわかった. 本報告では, 上記の問題点のうち, ①—③に関する改良方法について述べる.

2 走行加速度の向上

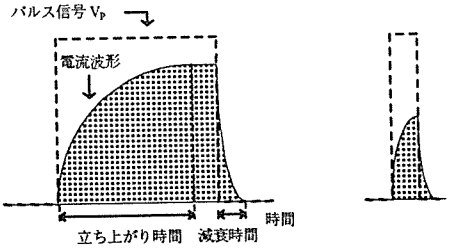
2000年3月の段階で, 本システムのロボットの走行性能について評価実験を行った際には, デッドレコニングによるロボットの直進時の距離誤差は約3%（移動速度 150mm/s, 移動距離 500mm), 片輪旋回時の角度誤差は約5%(移動速度50mm/s, 旋回角度 360°）であり, 低速時の走行誤差は少なく, 走行精度についてはまずまずの結果であった. しかし, 当時はモータ電圧が3.6V と低かったこともあり, 最高走行速度は, わずか 180mm/s に留まっていた[8].

そこで, 今回は, モータドライバを定電圧方式から定電流方式に変更することにより, 高速時のトルクの確保とあわせて, モータ消費電力の削減とを試みた.

2.1 ステッピングモータの回転数とトルク

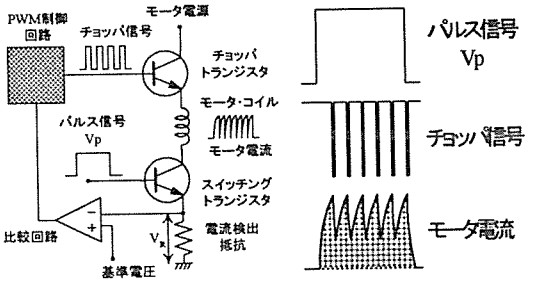
本ロボットでは駆動用にステッピングモータを使用している. 一般に, ステッピングモータの回転数はモータに送るパルス信号 Vp の周波数によって決まる. 回転速度を上げるには入力パルスの周波数をあげればよいが, モータ電流の立ち上がりの遅れにより, 十分に大きな電流が流れず（図1), トルクが減少する. モータを高速で回転させるためには, 電流を多く流し, 高速時のトルクを高めてやればよい.

従来, 本ロボットのモータドライバには定電圧ドライバを使用していた. 定電圧ドライバにおいては, 高電圧をかけて電流の立ち上がりを早くすることにより, モータ電流を多く流し, ある程度



(a) 低速走行の場合　(b) 高速走行の場合

図1　周波数によるモータ電流波形の変化(概念図)



(a) 回路　(b) 電流波形(概念図)

図2　定電流ドライバの動作原理

ロボットの走行速度をあげることが出来た．しかし，低速時には必要以上のモータ電流が流れ，消費電力が大きくなってしまうという欠点があった．

そこで，今回，モータドライバを定電流ドライバに変更し，モータに流れる電流を抑制することで，モータコイルに高い電圧をかけることを可能とした．同時に電流の減衰時間をカットすることにより，制動トルクを減らすことを試みた．

2.2　定電流ドライバの動作原理

今回，定電流ドライバの作成においては，モータにかかる電圧を細かく切断し，電圧の幅で電流を調整する“定電流チョッパ方式”を採用した[9]．この動作原理を図2に示す．図2(a)の回路では，モータに流れる電流を検出し，電圧 V_R と一定の基準電圧とを比較し，PWM制御回路により発生させたチョッパ信号のパルス幅でパルス信号を切り刻み，その幅を調節する．その結果，図2(b)に示すような波形のモータ電流が流れる．電流の量はチョッパ信号の幅でコントロールできる．

2.3　走行実験

定電圧ドライバを実装したロボットと定電流ドライバを実装させたロボットとに，同じ電圧(7.2V)をかけて以下の走行実験を行った．

2.3.1　実験手順

速度が上がるとトルクが低下するため，高速度を出すには低速から徐々に加速していく必要がある．そこで，走行速度を一定の値 α で予め定めた目標速度Vに達するまで加速することにした．今回は，目標速度Vを600mm/sから200mm/s間隔で設定していき，各速度での脱調しない最大の加速度 α を調べる．但し，加速スケジュールとしては，静止した状態から0.1sごとに α/10づつ速度を上げていくことにした．

2.3.2　実験結果

最大加速度 α の測定結果を表1に示す．定電流ドライバでは1100mm/sまで加速できたが，定電圧ドライバでは900mm/sまでしか加速できなかった．表1からわかるように，従来の定電圧ドライバでも，今回のように，3.6Vから7.2Vに電圧を上げ，徐々に加速することにより，最高走行速度を180mm/sから900mm/sへ向上させることができた．さらに定電流ドライバでは1100mm/sの最高速度を記録し，加速度 α を定電圧ドライバの約2～3倍にしても脱調せずに走行できた．したがって目標速度に達するまでの時間は1/2～1/3に短縮される．なお，最高速度1100mm/sには1秒で到達することができる．

表1　各ドライバでの最大加速度 α

(a) 定電圧ドライバ

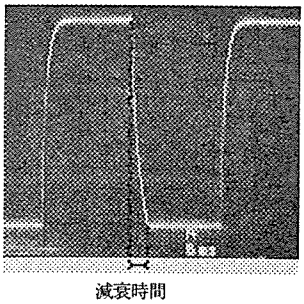
目標速度V (mm/s)	α (mm/s^2)
600	1200
800	570
900	450

(b) 定電流ドライバ

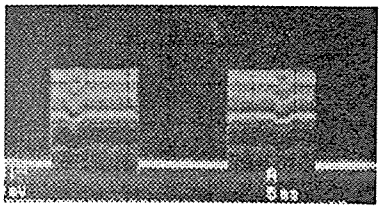
目標速度V (mm/s)	α (mm/s^2)
600	2000
800	2000
1000	1250
1100	1100

2.3.3　制動トルクの抑制

走行実験では，両ドライバともに同じ電圧(7.2V)をかけたにもかかわらず．定電流ドライバを用いた場合には，最高速度と加速度とが向上した．その理由は次のとおりである．パルスがOFFになった後もモータコイルにはしばらく電流が流れつづける．特にパルス周波数が高くなった場合(高速走行時)，電流の減衰時間の占める割合が多くなり(図3(a))，コイルに制動トルクが発生し，モータが回転しなくなる．一方，定電流ドライバでは減衰時間がカットされるように設計されている



(a) 定電圧ドライバの場合



(b) 定電流ドライバの場合

図3　モータドライバで観測されたモータ電流の波形．ただし，実際は電圧 V_R の波形を表示．

(図 2(a))．そのため制動トルクの発生が抑えられ，高速回転時のトルクが上がる．ドライバ内に流れる電流の波形を図3に示す．ただし，実際に図3に示したのは，電流検出抵抗のおける電圧 V_R（図2(a)）の測定値である．

3　経路計画

1章でも述べたように，昨年('00)までの本ロボット(KU-Boxes2000)の障害物回避は不十分であり，試合中にも他のロボットと衝突することが見られた．また，的確な経路計画アルゴリズムによって行動決定を行っておらず，目的地へ向けて必ずしも最短経路を走行しているとは限らなかった．そこで今回は，ロードマップ法に基づく，障害物回避と経路計画の方法について考察した．

3.1 仮想ロートマップ上での最短経路探索

移動ロボットの経路計画問題[10]においては，最小コストの経路を計画することがしばしば要求される．その方法の1つにロードマップ法（または，スケルトン法）がある[11]．これは，2次元の走行空間上に仮想的なノード・アークグラフ（ロードマップ）を設定し，ロボットの経路計画をそのグラフ上で立案すると言う，空間のスケルトン化の一手法である．ロードマップの一例を図6に示す．簡単のために，以下ではこの例を用いて，我々の経路計画法を説明する．

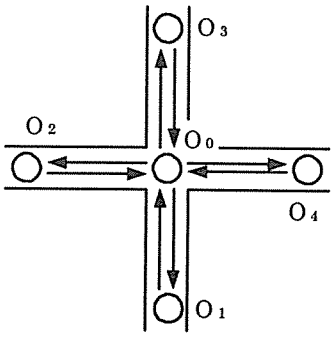


図4　展開前の交差点の表現

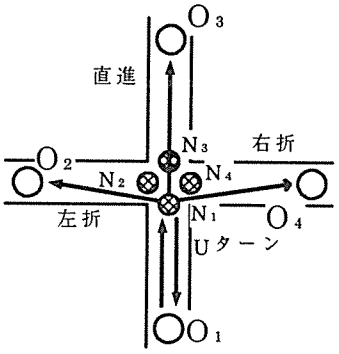
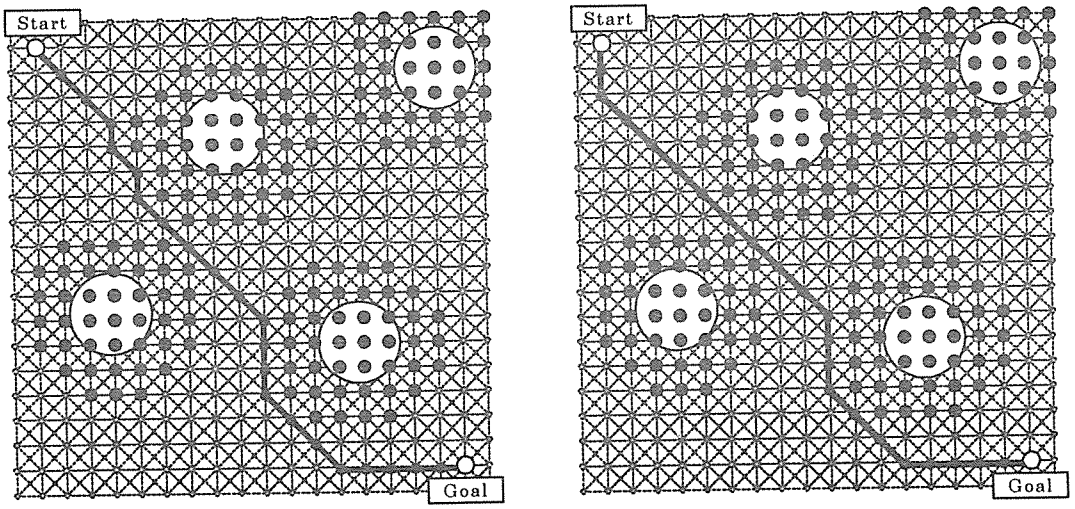


図5　交差点内ノード展開方式によるノードコストの表現

図6では，格子状の道路を考え，各格子点をノードとし，ノードを結んだアークをロボットの通行可能な経路とする．図6において，小さな○印がノード，それら2つを結ぶ線分がアークを表す．ここで水平・垂直方向のアークコスト（走行距離や走行時間に相当）を 1，斜め方向のアークコストを $\sqrt{2}$ とする．また障害物上にあるノードは使用不可能なノードとして設定し，それらを始点・終点とするアークのコストを無限大とする．図6において，大きな○印は障害物を表し，小さな●印のノードは通行不可能なノードを示している．なお，通行不可能なノードを障害物の大きさよりも若干大きめにとっているのは，ロボット自体の大きさを考慮したためである．このように，一旦，グラフのモデルができあがれば，グラフ上における任意の2点間の最短経路は，ダイクストラ法[12]によって容易に求めることができる．

3.2 ノードコストの表現

ダイクストラ法によって求められた2点間の最短経路は，現実のロボットが移動する際の最短時間の経路であるとは必ずしも限らない．なぜなら



(a) ノードコストを考慮しない経路 α　　(b) ノードコストを考慮した経路 β

図6　ロードマップ法を用いた経路計画

ば，現実のロボットが移動する経路では，グラフ上のノードにおける回転コスト（ノードコスト）を考慮する必要があるからである．我々は，こうしたノードコストを具体的に表現する方式を提案している[13]．ここでは，簡単のために 4 差路の交差点を例に，ノードコストを表現する方式について述べる．

通常行われている 4 差路の交差点の表現を図 4 に示す．本方式では，まず，交差点のノード O_0 を各方向に分解し，各方向に 1 個ずつノードを交差点の手前に新設する（図 5 のノード N_1〜N_4）．次に，新設したノードから各方向へ向けてアークを張る．図 5 では，ノード N_1 から各方向へ向けて張られたアークのみを記しているが，他の新設ノード N_2，N_3，N_4 についても同様である．次に新設されたノードから発するアークのコストにノードコストを加えて，それらのアークのコストとする．カーナビゲーションなどの応用で，実際の道路において，車両の右折や左折を禁止したい場合には，無限大のコスト値を加えればよい．

3.3 ノードコストを考慮した最適経路の例

この方式を用いてノードコストを考慮した経路計画の一例を示したのが図 6 である．図 6(a)は，ノードコストを考慮しない場合に得られた Start ノードから Goal ノードまでの最短経路 α である．経路 α は太字で示され，総コストは，26.97 である．

次に，ノードコストを考慮した場合の最短経路 β を図 6(b)に示す．ロボットが交差点で方向を転換するコストをノードコストとする．今回は，回転角に比例した大きさのノードコストを表 2 のように設定した．

表2　回転角度とノードコスト

回転角度	0°	45°	90°
ノードコスト	0	1	2

表 3 は経路 α と経路 β とについて，ノードコストを考慮する場合と考慮しない場合の総コストを示している．ノードコストを考慮しない場合の総コストは，経路 α，β ともに差はないが，ノードコストを考慮した場合の総コストは，経路 β の方が小さくなっている．これは，ノードコストを考慮した経路 β の方が，経路 α よりも総回転角度が少ないからである．

表3　各経路の総コスト

	ノードコストを考慮しない場合	ノードコストを考慮した場合
経路 α	26.97	33.97
経路 β	26.97	30.97

4 おわりに

今回，ロボットのモータドライバを定電圧方式から定電流方式へ改めることにより，高速走行時のトルクを高めることができた．これにより，従来の KU-Boxes ロボットの 2 倍以上の加速度で，ロボットを加速することが可能となった．現在のところ，1 秒で最高走行速度 1100mm/s に到達する．

なお，駆動モータの消費電力の削減に関しては，まだ，電流測定や耐久走行実験などを行っていな

いので正確な結論を出すことはできないが，体感として 1/2 以下には削減できたという感触を得ている．

また，本論文では，移動ロボットの走行制御に不可欠な経路計画問題に対しては，仮想のロードマップをフィールド上に設定することにより，障害物回避と回転コストを考慮した最短経路探索とを行う経路計画法を提案した．この経路計画法は，まだアイデア段階ではあるが，今後は競技用ロボットへの実装や，サッカー競技用以外の応用，例えば，無人搬送車の走行制御等への応用を行っていく予定である．

参考文献

[1] 小末将吾，五十嵐治一，黒瀬能聿，" RoboCup 小型部門用ロボットシステムーJP/S-II グループの現状ー" ('98.3　ホットトピックスと並列人工知能研究会資料 SIG-HOT/PPAI-9702，pp.1-3)

[2] 小末将吾，五十嵐治一，黒瀬能聿，" RoboCup 小型部門用ロボットシステムの開発ーJP/S-II グループー" ('98.4　第 3 回 JSME ロボメカ・シンポジア論文集，pp.21-24)

[3] 田中一基，朝岡忠，小末将吾，五十嵐治一，黒瀬能聿，" RoboCup 小型部門用共通ロボットシステムにおける画像処理サーバ - JP/S-II プロジェクト - " ('99.3 第 4 回 AI チャレンジ研究会，人工知能学会研究会資料 SIG-Challenge-9804，pp.1-4)

[4] 小末将吾，五十嵐治一，黒瀬能聿，田中一基，朝岡忠，" RoboCup 小型部門用共通ロボットシステムの開発 - JP/S-II グループ - "，第 4 回ロボティクスシンポジア予稿集 pp.87-92 ('99.3.30-31，仙台)

[5] 小末将吾，五十嵐治一，三吉孝則，飯土井修一，黒瀬能聿，"JP/S-II ロボットシステムの現状と問題点"，第 6 回 AI チャレンジ研究会資料 SIG-Challenge-9906，pp.58-63('99.10　生駒市)

[6] 五十嵐治一，小末将吾，田中一基，黒瀬能聿，五百井清，"JP/S-II プロジェクトの総括と今後の展開"，ロボカップスプリングシンポジウム 2000 ('00, 3 月，東京)，人工知能学会研究会資料 SIG-HOT/PPAI-9909, pp.1-5.

[7] http://www.fun.ac.jp/~ssuzuki/jopen2k/index.html

[8] 小末将吾，"グローバルビジョンを用いた分散協調型移動ロボットシステム"，'99 年度近畿大学大学院工業技術研究科修士論文，http://202.250.122.235/kenkyu2/masters/Kosue/Thesis.pdf

[9] 横山直隆："ステッピング・モータの制御法"，トランジスタ技術 SPECIAL, CQ 出版社, No.61, P132~ P133(1998).

[10] 藤村希久雄，"行動ストラテジーとアルゴリズム"，日本ロボット学会誌，Vol.11，No.8，pp1124-1129 (1993).

[11] J.C.Latombe, " Robot Motion Planning", Kluwer Academic Publishers, 1991.

[12] 茨木俊秀，福島雅夫，" 最適化プログラミング"，岩波書店，第 9 章，1991.

[13] 五十嵐治一，飯土井修一，"ノードコストを考慮した最短経路探索法とデータ構造"，近畿大学工学部研究報告，No.34, pp.77-80 (2000) (http://202.250.122.235/kenkyu2/masters/Iidoi/Kindai_TR_Iidoi.pdf)

ヘテロジーニアスチーム OZ における協調的行動の分析

Analysis on cooperations in the Heterogeneous RoboCup Simulation team Open Zeng

伊藤暢浩、西野順二、森下卓哉、久保長徳
Nobuhiro Ito, Junji Nishino, Takuya Morishita, Takenori Kubo

Open Zeng Project
bobson@phaser.elcom.nitech.ac.jp, nishino@se.uec.ac.jp

abstract

In this paper we show an analysis of cooporative teamwork of the heterogeneous RoboCup simulation team Open Zeng (OZ). The team OZ employs eleven different client program, for considering several realistic constraints such as difference and ill communication between clients. As a result, the team makes good match results in both JapanOpen 2000 and RoboCup 2000 Melbourne. Covering area analysis shows the reason of strength of OZ.

1 はじめに

本稿は、公募型ヘテロジーニアスクライアントチーム Open Zeng について、そのプレイの協調性を実際の試合結果に基づいて分析することを目的とする。また、こうした多様で多変数なマルチエージェントシステムの分析指針についても検討する。

チーム Open Zeng (OZ) は、2000 年から RoboCup シミュレーションリーグに参戦した、公募型の異種クライアントによる協調チームである [6, 3]。通常シミュレーションリーグにおいては、単一アルゴリズムのコピープレイヤによるチームになりがちである。OZ では作為的に、11 体のクライアントを互いに異なる原理とアーキテクチャで作成し、チームとして編成した。このようなヘテロ構成のチームが協調的かつ効果的にシステムとして機能するための要件について検討することを目的としている。

2000 年には日本国内大会のジャパンオープン函館と、世界大会である RoboCup in Melbourne にそれぞれ出場した。その結果たんなる寄せ集めではなく、チームとして有機的に機能して、上位の成績を記録した。

以下では、このヘテロチームの協調の様相について、メルボルン大会での試合結果とログを対象として、プレイヤ個々の行動とそれらの関係に基づいて分析を行い、チーム編成とチームとしての効果との関係について考察を行い今後の課題を明らかにする。

2 メルボルンでのチーム OZ

2.1 チーム編成

OZ は分散開発によるチーム [4] であり、非集中型の開発形態を取っている。11 体のプレイヤプログラムは、それぞれ別個の 11 人の開発者によって作られている。いっぽうで、各クライアントプログラムがどのようなパフォーマンスを持つか、また持つべきかといったチームとしての指針をあえて作らず、それぞれの開発者に完全に委ねている。このため各クライアントプログラムの詳細な仕様は、開発者以外のメンバーに取っては分からないものであった。他者の開発したクライアントに関する情報の入手は、実際にそのプログラムを起動して、その行動を各自で観察することが基本となっている。

チームとしての各クライアントの最小限の仕様は、それぞれのポジションとして与えている。これにより、各クライアントのロールと、最低限のチームとしてのまとまりを確保した。

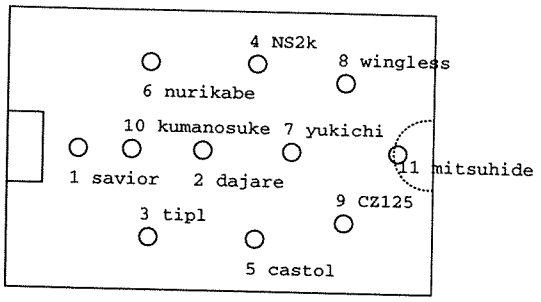


図 1: initial position

表 2: Melbourne result. '*' indicates OZ related player's teams

	opponent	goal-loss
GroupA	ATT-CMU	0-1
	A-Team	9-0
	Virtual Werder	4-2
Round 1A	11Monkeys *	1-0
Round 2A	YowAI *	0-1
Round 2B	Essex	0-4
	total	14-8

メルボルン大会でのチーム編成を表 1 と図 1 に示す。

図 1 に示した各ポジションを基準としているが、全てのプレイヤが必ずしもポジションどおりの働きをするとは限らない。それぞれのプレイヤの設計によって行動指針が異なり、あるプレイヤはポジションとは無関係にボールをつねにフォローしたり、また行動アルゴリズム上あまり活動しないプレイヤもいた。

2.2 試合結果

メルボルン大会における試合の結果を表 2 に示す。

予選を 2 位で勝ち上がり、決勝トーナメント 2 回戦で敗退した。

OZ に含まれる各プレイヤの本来のチームとの対戦では、OZ の方が負けていることも勝っているもあることから、いちがいに連合チームにすることによって一方的に強くなったり、弱くなったりするものでは無いことが分かる。

また上位の成績のチーム相手にも勝利および善戦していることから、個々のプレイヤの能力の和だけでなく、その編成方法も、チームとしての能力に大きく影響することが確かめられた。

3 作業空間での協調行動の分析

チーム全体としての協調プレイの様相を分析するために、サッカーフィールド平面を作業空間とみなし、プレイヤの行動領域を調べた。

予選で互角に戦ったチーム Virtual Werder との試合結果について、11 体のプレイヤ個々の行動を図 4〜図 14 に示した。ゲーム全時間 6000 時間の移動軌跡を表示している。これによって、各クライアントの行動域、静的なポジションと、基本的なアルゴリズムが分かる。

3.1 カバリング領域による協調の分析

以下で分析を進めるために、カバリング領域を定義する。カバリング領域とは、各プレイヤの行動軌跡を包含する領域と、それらの和で表される領域である。これは 6000 ステップの全時間を畳み込んだ静的な行動領域の指標となる。

全てのクライアントのカバリング領域として、軌跡全体の和を図 2 に示した。これは図 3 に示した、ボールが移動した領域全体をカバーしている。ここで、もしカバーされていない部分があるとすると、チームとしてその領域でのボールコントロールを得ることができないことを示すことになる。

今回の結果では、各プレイヤの移動領域によるカバーが行われ、チーム協調を発現する必要条件を満たしていると言うことができる。ただしこれは、時間を畳み込んだ静的な範囲での分析であり、時間軸なども考慮したコンフィギュレーション空間で位相的に到達できないボールとプレイヤ配置が発生しないことを示すものではない。

3.2 各プレイヤの分析

各プレイヤの行動アルゴリズムについては、 各チーム個々の報告 [2, 1, 5] に掲載されている。ここ

表 1: a Formation of Open Zeng; abbreviations are FW:=Forward player, MF:=Mid field player, DF:=Defense player, GK:= Goalie, Fukui: Fukui University, JAIST: JAPAN Advanced Institute of Science And Technology, Keio: Keio University, Kinki: Kinki University, Mie: Mie University, NITECH: Nagoya Institute of Technology, TITECH: Tokyo Institute of Technology, UEC: The University of Electro-Communications

Position	Player name	developer name	affiliation	Orig. team	code
FW	mitsuhide	Suzuki T.	UEC	YowAI 99	C
	CZ125	Koto T.	UEC	TakAI	C++
	yukichi	Kinoshita S.	Keio	11Monkeys 99	C++
		Tanaka.N.		11Monkeys 00	C++
MF	Tipl	Shinoda T.	JAIST	Kakitsumabata	C++
	wingless birdie	Morisita T.	Fukui	Zeng	C++
DF	Dajarenja	Igarashi H.	Kinki	KU-Sakura	C
	Kumanosuke	Hioki T.	Mie	Sfidante	C
	Bob	Ito H.	NITECH	Kakitsubata	Java
	NS2k(ver.haken)	Nakagawa K.	NITECH	NITStones99	Java
		Esaki T.	NITECH	NITStones00	Java
	nurikabe	Kubo T.	Fukui	gnez	C
	Castol	Oota M.	TITECH	gemini	C
GK	savior	Shimora H.	Fukui	Zeng	C++

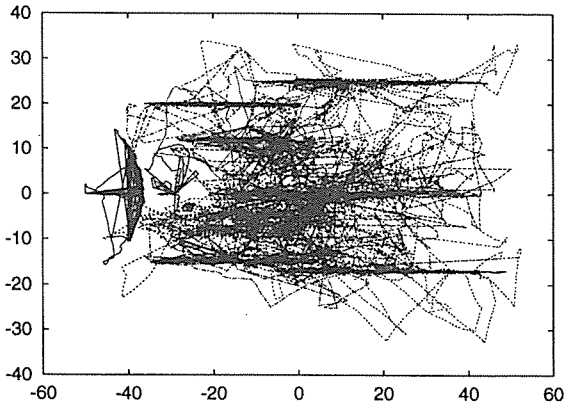


図 2: all players coverage

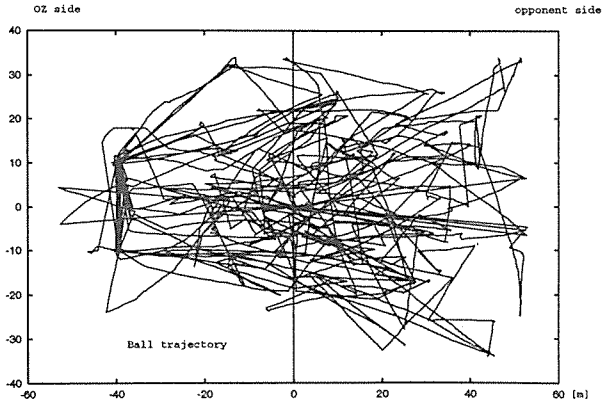


図 3: Ball trajectory area

では、試合結果からの分析を行う。

強力なゴーリーである Savior は、特徴的な行動を取っている。図 4 で分かるようにペナルティエリア付近で、ゴールを中心とした円弧を描いて移動している。ボールを持つ相手フォワードと 1 対 1 の時に、ゴールを背にして立つという人間が用いているヒューリスティックが実現されている。

他のフィールドプレイヤの多くは、いわゆるタテの移動をしていることが図 2 から明らかである。各図では x 軸方向に直線状の軌跡が濃く重なっていることが分かる。ボールや他のプレイヤとの関係から、DF(ディフェンダー) も FW(フォワード) もそれぞれにオフサイドラインを意識した行動をしていることを示している。全て異なる開発者であるが、OZ に参加したクライアントの属する元チームがそれぞれ、オフサイドラインに敏感な行動を基準としていることが分かる。

また、これらのクライアントは、タテの動きをもとに、必要に応じて左右 (図では上下) に移動し、ボー

ルへのアプローチを行っている。逆にいえば、ボールから遠いときには、フィールドのタテのポジションライン付近にそれぞれ戻っていることが読み取れる。

3.3 特徴あるプレイヤとチーム連係

特徴的なのは、１０番の kumanosuke、２番 dajare、７番 yukichi の３体の行動とその連係である。

kumanosuke は、図 13 で分かるように、初期ポジションからほとんど移動していないという個性的な行動をしている。これはクライアントが極力行動しないという方針で作られているためである。クライアント自体が本来発揮できるパフォーマンスと比較すると、機能性は低いプレイヤであると言うことができる。しかしながら、全体を示す図 2 で見ると分かるように、このプレイヤの行動域とちょうど他のプレイヤのカバー域とがかみあい、チームとしては問題が無かったことが分かる。他のプレイヤが kumanosuke 領域に立ち入らなかったのは、ポジションのためと、他のプレイヤが kumanosuke の存在を見て、スタミナ保持などからそばによらないように作られていたためと考えられる。つまり、今回の kumanosuke の行動領域の大きさは、他のプレイヤのスタミナ消費によってカバーされてはいたが、その負担は許容量のうちであったと言うことができる。逆に言えば、カバープレイヤのスタミナ外でのカバーが必要であったなら、制御不能領域が発生してしまったかも知れない。

dajare は、典型的なボール追従型プレイヤである。本来のポジションである中央地点は単なる始点であって、試合中は可能な限り移動している。このため、ボールが早いパスで移動してしまうと、戦線から離脱して戦力外となる場面も多かった。いっぽうで相手チームともみあっている間にボールに追い付き、ボール周辺での仲間プレイヤを増やしボールコントロールが有利にするという効果があった。

yukichi は、動的な役割を持つプレイヤによるチーム 11Monkeys からのプレイヤである。行動軌跡において他のプレイヤ同様のタテ移動を中心としつつも、他のプレイヤと比較して広範囲まで移動している。このことからも動的な役割を持ち、必要な場面ではポジションを崩してボール周辺での優位性を増すという行動が分かる。とくに、フォワード付近ではセンタ・サイド両域で攻撃に参加していること、左サイドバックでもディフェンダーとして行動していることが分かる。

全体としては、ボール移動領域に対して抜けのないフィールドカバリングができていた。このことがバランスの良いチームとして実際に良い結果を導いていると考えられる。

しかしながら、このカバリングは計画された物ではなく、結果として現れたものである。とくに、kumanosuke 周辺では移動軌跡の領域カバリングがまばらになっている。抜けが無かった理由は、kumanosuke の直前が、広移動型クライアントの dajare であり、また、その前には高機能クライアントの yukichi がいて、動的にカバーされたためである。このため今回は抜けが無かったが、これはある種の偶然であり、チームを編成するときの最低条件としてあらかじめ検討しておく必要がある。

3.4 オフカバリング領域とその解消

全体図 2 で分かるように、左下の部分は行動カバリングから外れている。これはボールがその付近に来なかったことにもよるが、右サイドのプレイヤが全て左サイドに比べて中央寄りを規定ポジションとして取っていることによることは図 2 から明らかである。今回の相手は、ラインぎわの深いサイドを使わないチームだったため、この部分のオフカバリングが問題とならなかった。他のチーム相手の試合ではこの部分を攻撃され不利になっている場面も存在する。

いっぽうこのような左右の非対象性は、ヘテロチームの特徴である。プレイヤをモデリングするようなチームが相手の場合には、こうしたプレイヤ毎の個性がある方が、学習のしやすさを減らすという意味で有効となるだろう。

また savior のように一定の曲線しかうごかなくとも、敵ボールを確実に止めるプレイヤ方式もある。これはボールキャッチというゴーリーの特殊性も生かし、かつ敵との位置の相対関係からゴールそばでのボール領域、いわゆるシュートコースを制限していることによっている。行動によってボール領域を間接的に制御して、結果として狭いカバリング領域ながら十分なカバーを行っている。

あわせて考えると、個々のプレイヤの持つべきカバリング領域を単に広げるのではなく、相手に応じたボールコントロールでボール領域を狭め、そのうえで必要なカバリング領域を作りだして重ねるということが、チームとして設計することが望ましい。

4 まとめ

プレイヤとボールの移動領域をそれぞれカバリング領域として定義し、そのかさなりによって OZ の試合結果を分析した。その結果対 VW 戦ではチームとして有機的に行動することができ、結果として勝利できたことが分かった。

プレイヤ個々を詳細に分析することなく、全体的な指標を提案しヘテロなチームの効果を判断する手法を提案できた。自分チームで 11 体、相手とボールもいれて 22 体の運動についての詳細な分析は計算量的に考えて相当に困難である。このような方法でチーム分析ができれば、OZ のようなヘテロなチームはもちろん、通常のチームの分析にも応用できると考える。

今後の課題としては、実際には時系列での分析や、プレイヤどうしの相関についても、位相的な分析を行い、系統的なオフカバリング領域の発見方法が必要である。さらにそのオフカバリング領域解消のための効果的な方策、どのプレイヤをどのように修正するべきか、などを与える手法の開発も望まれる。

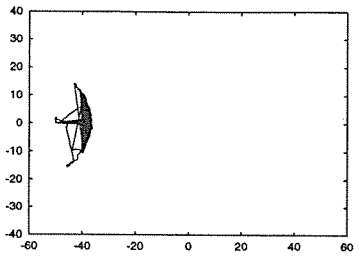


図 4: Savior

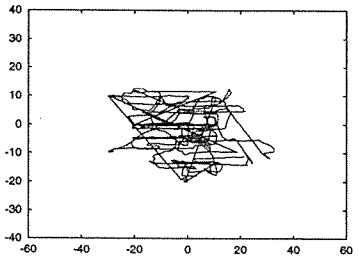


図 5: Dajarenja

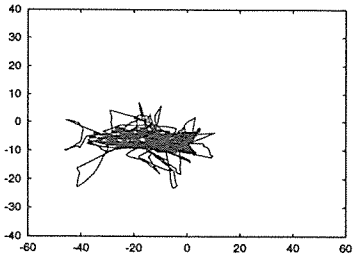


図 6: Tipl

参考文献

[1] M. Asada and H. Kitano, editors. *RoboCup-98: Robot Soccer World Cup II.* Springer, 1999. ISBN 3-540-66320-7.

[2] H. Kitano, editor. *RoboCup-97: Robot Soccer World Cup I.* Springer, 1998. ISBN 3-540-64473-3.

[3] J. Nishino, T. Morishita, and T. Kubo and. *RoboCup-00: Robot Soccer World Cup IV*, chapter Open Zeng: An Open style distributed semi-cooperative Team development project from Japan, page to be appeared. Springer, 2001.

[4] Junji Nishino. http://bishop. fuis. fukui-u. ac. jp /~nishino /oz/.

[5] M. Veloso, E. Pagello, and H. Kitano, editors. *Robocup-99: Robot Soccer World Cup III.* Springer, 2000. ISBN 3-540-41043-0.

[6] 西野順二, 森下卓哉, 木下修平, 鈴木隆志, et al. シミュレーションドリームチーム oz の挑戦. In *AI チャレンジ研究会第 9 回資料*, pages 16–19. 人工知能学会, 2000.

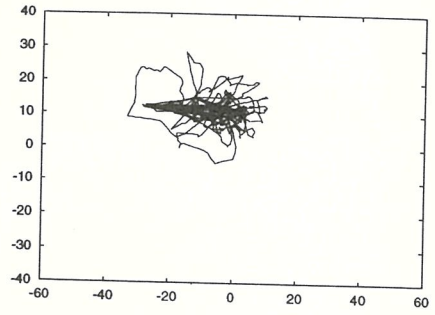


図 7: NS2k

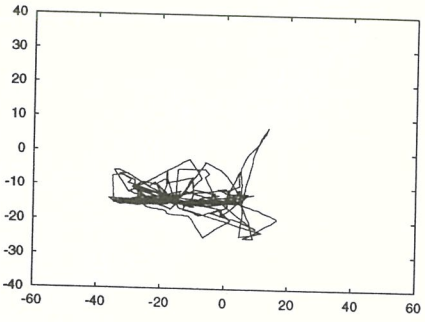


図 8: Castol

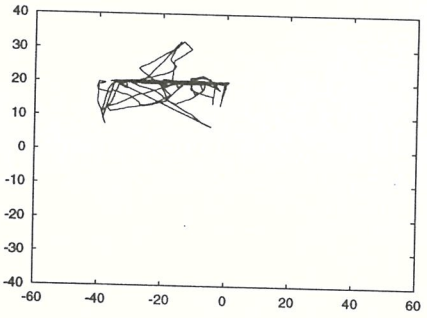


図 9: nurikabe

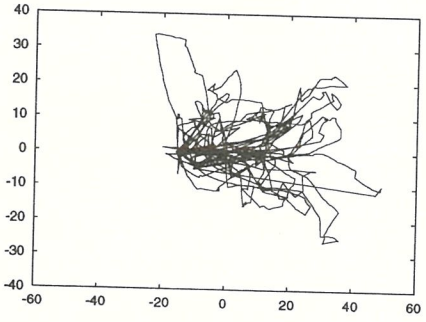


図 10: yukichi

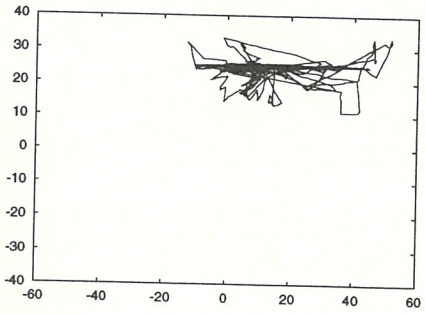


図 11: wingless

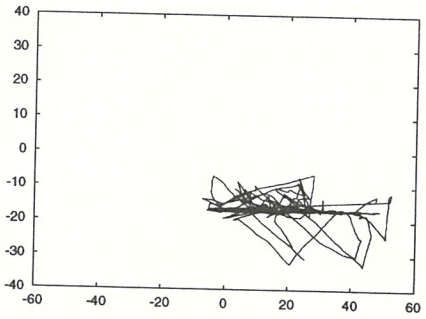


図 12: CZ125

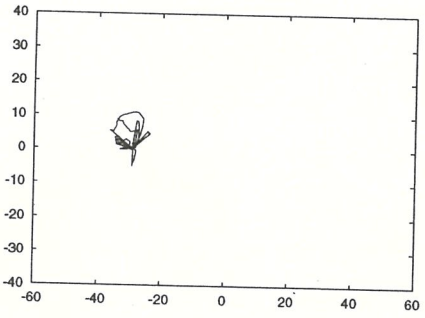


図 13: kumanosuke

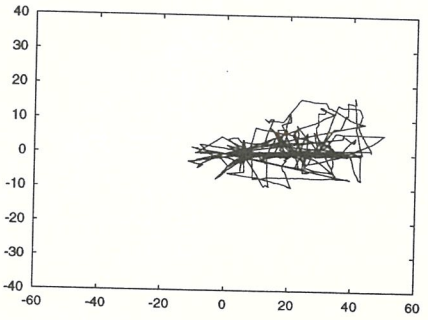


図 14: mitsuhide

〒162 東京都新宿区津久戸町 4-7 OS ビル 402 号室 03-5261-3401 Fax: 03-5261-3402

(本研究会についてのお問い合わせは下記にお願いします.)

AI チャレンジ研究会
主 査
奥乃 博
京都大学大学院 情報学研究科
知能情報学専攻 音声メディア分野
〒606-8501 京都市左京区吉田本町
075-753-5376 Fax: 075-753-5977 /
科学技術振興事業団 ERATO
北野共生システムプロジェクト
okuno@nue.org

担 当 幹 事
浅田 稔
大阪大学大学院 工学研究科
知能・機能創成工学専攻 創発ロボット工学講座
〒565-0871 大阪府吹田市山田丘 2-1
06-6879-7349 Fax: 06-6879-7348
asada@ams.eng.osaka-u.ac.jp

Executive Committee
Chair
Hiroshi G. Okuno
Dept. of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University
Yoshida-honmachi Sakyo-ku, Kyoto, 606-8501, JAPAN /
Kitano Symbiotic Systems Project, ERATO, JST

Secretary in Charge
Minoru Asada
Dept. of Adaptive Machine Systems
Graduate School of Engineering
Osaka University
2-1 Yamadagaoka, Suita, Osaka 565-0871, JAPAN

SIG-Challenge web page; `http://www.symbio.jst.go.jp/sig-challenge/`