平成 13 年度

学士学位論文

RoboCup エージェントの下位行動の性能向上

Ability Progress of Low-Level-Behavior in RoboCup Agents

1020279　　川口 宏

指導教員　　Ruck Thawonmas

2002 年 2 月 8 日

高知工科大学 情報システム工学科

要 旨

RoboCup エージェントの下位行動の性能向上

川口 宏

本稿では，RoboCup エージェントの下位行動の性能向上について述べる．下位行動は，ドリブル，パスといった基本行動のことである．下位行動の中のスマートドリブルは，Patrick Riley によって提唱された．これをエージェントに実装し，敵エージェントに対する回避行動の性能評価を行った．その結果，今までのエージェントよりも性能の向上に継った．

キーワード RoboCup，エージェント，下位行動，スマートドリブル

Abstract

Ability Progress of Low-Level-Behavior in RoboCup Agents

Hiroshi KAWAGUCHI

This thesis describes about ability progress of low-level-behavior in RoboCup agents. Low-level-behavior are dribble and pass that is basic behavior. Smart-dribble in low-level-behavior is proposed by Patrick Riley. This was mounted in the agent, and ability evaluation of the evasion action to an enemy agent was examined. Consequently, compared with the former agent, that ability progressed ability.

key words RoboCup, agent, low-level-behavior, smart-dribble

目次

目次

図目次

第 1 章

はじめに

高知工科大学情報システム工学科では,「 RoboCup シミュレーションを通じてのソフトウェア工学及び人工知能の教育 」 として，RoboCup シミュレーションリーグを使った授業が行われている．この RoboCup とは，人工知能と知能ロボットに関する研究を促進するために始められた，サッカーを題材にしたプロジェクトである．

現在，高知工科大学情報システム工学科 RoboCup チームの基本性能の現状は，ボールを蹴って追い付いて蹴るという，最低限のドリブル，そして，パスは受け手とパスする側が，ボールの軌道を予測出来ていないということもあり，事実上成立していない．

このようなエージェントの性能を左右する基本行動が確立されていないという問題が，高知工科大学情報システム工学科の RoboCup のチームにはある．これにより，チーム戦略・プランニングが立て辛くなっている．

この問題を解決するため，まず RoboCup 世界大会で優勝したチームのプログラムを参考にし，高知工科大学情報システム工学科のチームに実装する．そのうえで，エージェントの基本性能を向上させることが，本研究の目的である．

第 2 章

RoboCup とは

本章では，はじめに RoboCup 全体について述べる．次に本稿で扱う RoboCup シミュレーションリーグについて説明する．

2.1 RoboCup の歴史と目的

RoboCup は，人工知能と知的ロボットに関する研究を促進するためのものである．自律移動ロボットによるサッカーを題材として，日本の研究者たちによって 1995 年に提唱された．現在では，サッカーだけではなく，大規模災害シミュレーションや災害救助ロボットの研究・開発を行なう，RoboCup レスキュー [1] などがある．

RoboCup サッカーには，現在 4 つのリーグがある．

- シミュレーションリーグ

  コンピュータ上の仮想フィールド上で，1 チーム 11 体のソフトウェアエージェント同士によって行なわれるリーグ．RoboCup サッカーの中では一番古くから存在するリーグで，一番洗練されたチームプレイをする．本稿ではこのリーグを扱う．
- 小型リーグ

  卓球台とほぼ同じ大きさのフィールドで，直径 18 cm 以内に入る小さなロボットが 5 台以内でチームを組み，オレンジ色のゴルフボールを使って対戦するリーグ．
- 中型リーグ

  卓球台 9 枚分の大きさのフィールドで，直径 50 cm 以内に入るロボット 4 台でチームを組み，オレンジ色のフットサルのボールを使って対戦するリーグ．

- Sony 四脚ロボットリーグ

  SONY のエンターテインメントロボットによる 3 対 3 で行うリーグ．このリーグは，選抜されたチームに貸与されたロボットで競技を行うため，一般参加は出来ない．

さらに，2002 年にはヒューマノイドリーグが加わる予定になっている．

毎年 RoboCup では，国内大会と世界大会が行なわれる．ここで，各チームの研究・開発されたロボット同士が勝負をしている．

RoboCup サッカーの最終目標は，ヒューマノイドリーグのチームが 2050 年に FIFA のワールドカップの優勝チームに勝つことである．

2.2 RoboCup シミュレーションリーグについて

2.2.1 RoboCup シミュレーションリーグの概要

RoboCup シミュレーションリーグは，RoboCup の中で一番古くからあるリーグである．そのシミュレーションの仕組みは，サーバ・クライアント方式を採用している．この方式を簡単にまとめた図を図 2.1 に示す．

サッカークライアントとサッカーサーバの間は，UDP / IP 通信によって通信が行なわれる．よって，サッカークライアントを作成する場合には，UDP / IP 通信をサポートするプログラム言語であれば何でも良い．本稿では，Java 言語を使用している．

また，1 つのクライアントは原則として，1 つのエージェントのみを制御しなければならない．1 つのクライアントが複数のエージェントの情報を得て，行動を集中制御することは許されていない．

そして，サッカーサーバによってシミュレートされた結果は，サッカーモニタと呼ばれる表示プログラムによって表示される．

現在のサッカーサーバの最新バージョンは 8.02 (平成 14 年 2 月 4 日現在) である [2]．

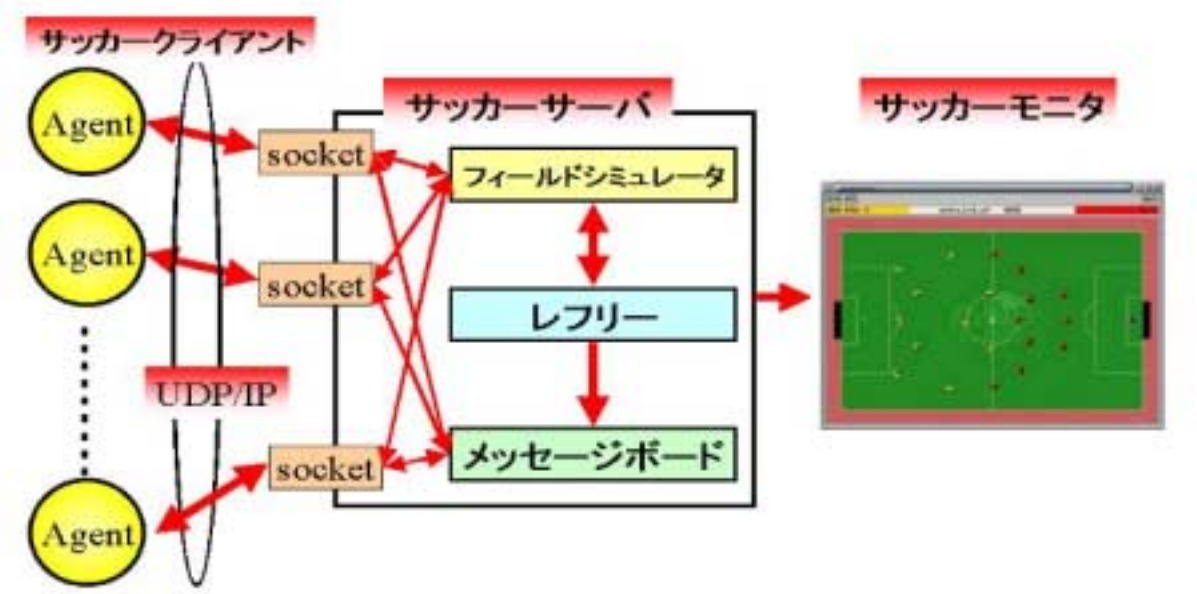


図 2.1 シミュレータの構成図

2.2.2 サッカーサーバの仕組み

サッカーサーバは，サッカーフィールド，ボール，審判，プレイヤーの位置などをシミュレートする．サッカーフィールドとフィールド上の物体は，平面で扱われ，高さの情報は持っていない．また，プレイヤーとボールは円として扱われる．動きの情報は，1 シミュレーションサイクル（100 msec）毎に離散的に更新される．サイクルの終わりに，サッカーサーバは受信したすべての行動コマンドをフィールド上のすべての物体に適用して，次のサイクルの位置と速度を計算する．

シミュレーションには，現実世界に近づけるために予期せぬ動きをするように，ノイズと風の情報が加えられる．

2.2.3 エージェントの行動

エージェントは次の行動をすることが出来る．

- turn　エージェントの向いている方向を体ごと変える．
- turn_neck　エージェントの向いている方向を頭だけ変える．体の方向は変わらない．
- dash　エージェントの体が向いている方向に加速をつける．
- kick　ボールが蹴れる範囲内（ kickable_area ）にある場合に，方向と強さを決めてボールを蹴る．
- move　得点が決まったとき等の初期フォーメーションに戻るためと，ゴールキーパーが catch した時，ゴールキック位置に移動する時のみ使えるの行動．
- catch　ゴールキーパーがボールをキャッチする行動．ゴールキーパー以外やペナルティーエリア外では無効．
- say　メッセージをすべてのエージェントに伝える．
- change_view　視野角度と視覚情報の品質を変える．これを変えると視覚情報の送られてくる頻度が変わる．

サッカークライアントは，これらの行動を駆使して設計しなければならない．原則として 1 サイクルに 1 つの行動しかできない．ただし，turn_neck，say，change_view は，ほかのコマンドと同時に 1 サイクル中に送ることが出来る．

また，エージェントにはスタミナがある．スタミナは，行動を起こすと減少する．減少する割合は，行動の度合い（ 全力で走る等 ）で変化する．スタミナが少ないと全力で走れないなど等の制限が加わる．行動をしなければ，スタミナは少しずつ回復する．

2.2.4　センサ情報

サーバからセンサ情報としてクライアントに次のような情報が送られてくる．

- 視覚情報　クライアントの視野内に入ったオブジェクトの情報．
- 聴覚情報　審判からの判定メッセージと，クライアントプログラムが say コマンドで送ったメッセージ情報．
- 感覚情報　自プレイヤーのスタミナ，視野モード，速度，頭の向き，kick, dash, turn,

say, turn_neck のコマンドを送った回数の情報.

センサ情報がサーバから送られてくる頻度は，それぞれ違う.

視覚情報は，初期状態で 150msec 毎にクライアントに送られる．ただし，シミュレーションサイクルとは非同期である.

聴覚情報は，メッセージが発生したときにシミュレーションサイクルとは非同期に送られる.

感覚情報は，シミュレーションサイクル毎に必ず送られてくる.

第 3 章

KUT RoboCup Project について

3.1 KUT RoboCup Project とは

高知工科大学情報システム工学科では,「 RoboCup シミュレーションを通じてのソフトウェア工学及び人工知能の教育」と題して，RoboCup を使った授業・研究を 1999 年度より行っている [6]．KUT RoboCup Project の概要図を図 3.1 に示す.

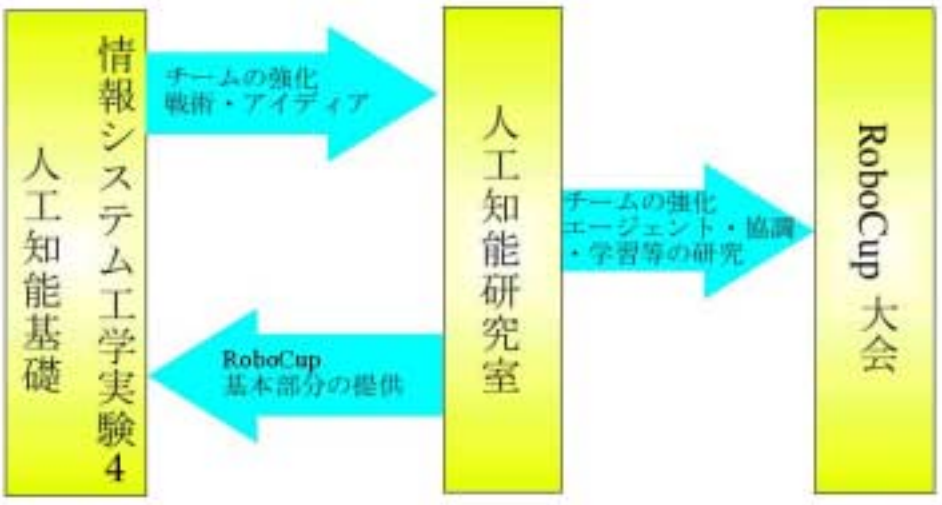


図 3.1　KUT RoboCup Project 概略図

図 3.1 で示したように，人工知能研究室と授業 (情報システム工学実験 4 (以下，実験 4

)・人工知能基礎ともに 3 年次) で相互協力し，年 1 回の RoboCup 大会に代表チームを送り込むことが目的である．実験 4 では，実際にクライアントを作ってもらう．最終的には大会を行い，戦術や機能のアイデアを出し実装してもらう．また，人工知能研究室の役割としては，次の 3 つがある．1 つは，実験 4 で使用するクライアントプログラムの基本部分を開発・提供を行う．2 つ目は，実験 4 で出されたアイデアや，戦術，機能，などを統合し，代表チームの強化に利用する．3 つ目は，人工知能に関する研究の成果を代表チームに追加する．

Project 初年度の 1999 年度の実験 4 のチームは，人工知能研究室で強化され，2000 の RoboCup Japan Open に出場した．結果は，初出場ながら予選リーグを突破し決勝トーナメントに出場することが出来た．今後，この Project によって強化されたチームで上位入賞を目指す.

3.2 RAIK-NTG4 とは

RAIK-NTG4 は，今年度の高知工科大学情報システム工学科のチームプロジェクトで，RoboCup サッカー世界大会へ向けてのソフトウェアエージェントチームの開発プロジェクトである．RAIK-NTG4 の図 3.2 を示す.

図 3.2 の説明.

- HAM

  HAM とは，強化学習によって行動を決定していくエージェント.

- Huma

  Huma とは，人の意志決定挙動を適用したエージェント.

- ハンドコーディング

  ハンドコーディングとは，従来からの手法で作成したエージェント.

- SemiReal / Original Monitor

  SemiReal / Original Monitor とは，エージェントの得た情報を視覚的に表現する独自

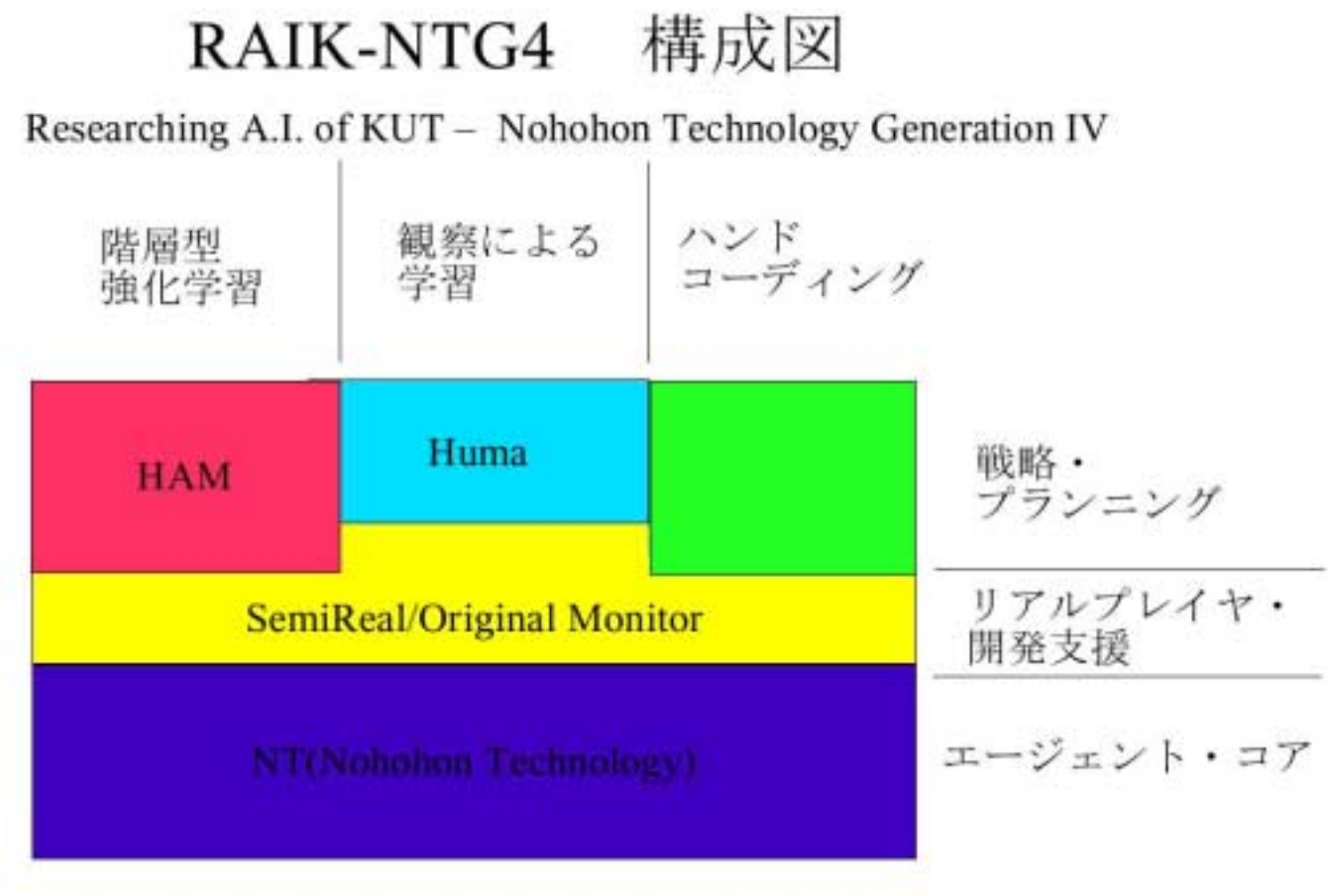


図 3.2　RAIK-NTG4 の構成図

のモニタ．また，これはエージェントを操作できるようになっている．

- NT（ Nohohon Technology ）

  NT（ Nohohon Technology ）とは，RoboCup シミュレーションクライアントとしての，基本機能（ サーバとの情報の送受信，情報の保存と処理 ）を備えた部分．

3.3　RAIK-NTG4 における本研究の位置付け

本研究は，主にエージェント・コアの部分（ NT ）に位置する．エージェント・コアは，エージェントの基本となる部分で，エージェントの行動を決定する部分である．

3.3 RAIK-NTG4 における本研究の位置付け

本研究では，エージェントの行動を司る部分の研究であるので，サーバとの情報の送受信，情報の保存と処理については触れていない．

そのエージェントの行動について，RoboCup シミュレーション部門の世界大会で優勝したチームのプログラムを参考にして，高知工科大学情報システム工学科のチームに実装する．

第 4 章

下位行動の向上法

4.1 下位行動の種類

下位行動には，ドリブル・パスがあり，そして，ドリブルの発展型にスマートドリブルがある．

4.1.1 ドリブル

ドリブルの基本的な方法はとても簡単で，kick と dash を繰り返すだけである．ボールが kickable-area にあれば蹴り，無ければボールを追いかけるというものである．これを図 4.1 に示す．

理論的には単純だが，実際にはボールを蹴るときにはボールとの距離や角度によってボールを蹴る強さが変わるので強く蹴りすぎて敵にボールを取られたり，弱すぎて進むスピードが遅くなったりと，うまくドリブルをする事は難しい．

そして，ドリブルの行動は常に 2 サイクル後を考えて，エージェントが動くべきだという事である．その理由については，dash コマンドと kick コマンドは 1 サイクルにどちらか片方しか送れないので，次に dash する時は前に dash した時の 2 サイクル後となっているからである．さらに，もし，kick の方向がエージェントの進行方向で無かったなど turn しなければならなくなった時は 3 サイクル後になるからである．このようなことがあるために，次のスマートドリブルが考え出された．

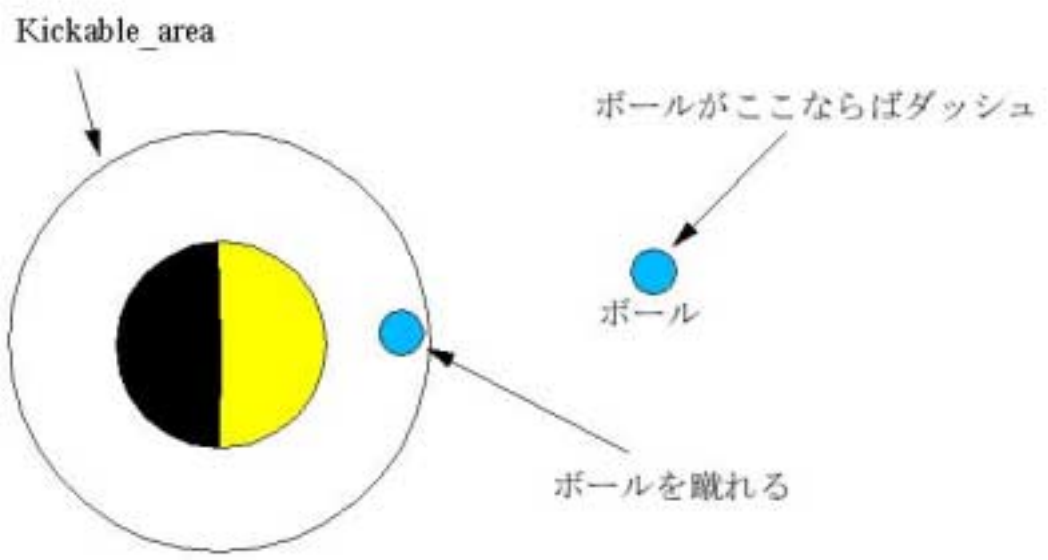


図 4.1　ドリブルの図

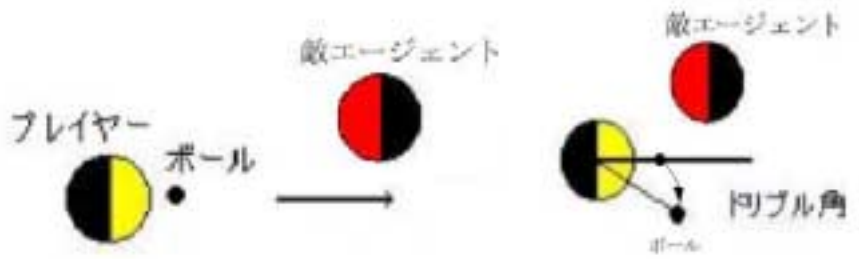
4.1.2　スマートドリブル

スマートドリブルは，Carnegie Mellon univercity の Patrick Riley 氏が提唱した技術である．この技術を使い Carnegie Mellon univercity は，’98，’99 年と RoboCup 世界大会で優勝を治めている．

スマートドリブルの基本はドリブルと同じである．スマートドリブルでは，ドリブル角とそれに対する重み付けというものを使って，敵エージェントを避けるというドリブルである．

エージェントは感覚情報から敵エージェントの情報を取得し，この情報から自分はどの方向にドリブルすればよいかを考えドリブル角を決定する．考えられる選択肢に敵エージェントの位置によって重み付けを変えていく．

例えば，相手が自分から見て 45 度の角度にいるならば，-90 度の角度に 1 票入れるといった方法で行われる．近くにいる敵エージェントが 1 人の場合はこれで良いが，複数いる時は相手との距離や人数によって重みの付け方が変わる．これを図 4.2 に示す．



(a) 現在の位置　　(b) 2 サイクル後の位置

図 4.2　スマートドリブルの図

4.2　下位行動の特徴

この節では，ドリブルとスマートドリブルについての特徴を述べる．

4.2.1　ドリブル

ドリブルの利点というものは，ドリブルすることの難しさを知る上で最も重要な部分である．そして，エージェントの基本となっているので，きちんと kick と dash をなされていないとエージェントの質が落ちるということもあります．

そして，欠点は，ドリブルはただ kick と dash を繰り返すだけなので，ボールを敵エージェントに取られやすいということがあげられる．

4.2.2　スマートドリブル

スマートドリブルの利点は，普通のドリブルの欠点を補ったもので，敵エージェントを避けるようにボールを kick するので，敵エージェントにボールを取られにくくなっているということである．

そして，ボールの位置を予測して，kick しているので，ボールを見失うことも少なくなっている．また，味方チームがボールをもっている時間が長くなるので，戦略プランが立てやすくなるということもあげられる．

欠点としてあげられることは，ドリブルのスピードが遅くなってしまうことである．なぜなら，スマートドリブルは，普通のドリブルと比べてより多くの kick をするからである．普通のドリブルは，ただ前にボールを蹴るだけなので，ドリブルのスピードは一定だが，スマートドリブルでは，その特性ゆえにドリブルのスピードが遅くなってしまうという欠点がある．

また，現段階ではドリブル角の重み付けをし，ボールを移動させせいる間は，自エージェントは止まっているという欠点がある．これは，止まっていては一体の敵エージェントならば問題は無いが，実際は一体でサッカーをするのではないので，別の敵エージェントにボールを取られてしまうということがあるためである．

欠点の方を多く記述したが，実際は利点の方が欠点よりも大きいため，RoboCup Japan Open などの大会において，上位チームは必ず取り入れている技術でもある．このことからも分かるように，スマートドリブルは重要性の高い技術である．

4.3 アルゴリズム

4.3.1 ドリブル

ドリブルをするためのアルゴリズムは，主に式 (4.1)，(4.2) を使用する．

$$Pnew = Pcurrent + v + (v * pdecay + a) \tag{4.1}$$

$$traj = \frac{Ptarget - Pball}{1 \quad + \quad bdecay} \tag{4.2}$$

式 (4.1) は，エージェントの 2 サイクル後の位置を計算するもので Pnew が予想されるエージェントの位置で，P$cuurent$ は現在のエージェントの位置，v は現在のエージェントの速度，$pdecay$ はサーバーパラメーターの player_decay，そして a が dash コマンドで与えられた値である．しかし，これは通常の状態で，エージェントが疲れているとき (stamina が減っているとき) はそれを考慮に入れる必要がある．

式 (4.2) はボールの 2 サイクル後の位置を計算する式で，$traj$ というのが図 4.3 のボールとボールを結ぶ矢印になる．P$target$ がボールの目標位置，P$ball$ が現在のボールの位置であり，$bdecay$ はサッカーサーバーパラメータの ball_decay である．目標位置 P$target$ (例えば絶対座標で (X , Y)) に蹴ろうと思えば式 (4.2) で与えられる $traj$ (x , y) の位置から，kick コマンドに渡すパラメーターを考える必要がある．

以上二つの式から dash 後のエージェントの kickable-area 内にボールがあるように dash 及び kick コマンドに渡す値を求めることが必要になってくる．

さらに「ドリブル角」と呼ばれるものを考慮に入れる必要がある．ドリブルの目標地点はこのドリブル角の有効範囲 [-90 , 90] に収まるように決めるのが良く，ドリブル角が有効範囲を超えるとうまくドリブルできなくなる．

ドリブル角がどの角度をなのかについては図 4.3 に示す．ドリブルする時のボールの位置は自分よりも前にあるべきという事を考慮することが重要である．

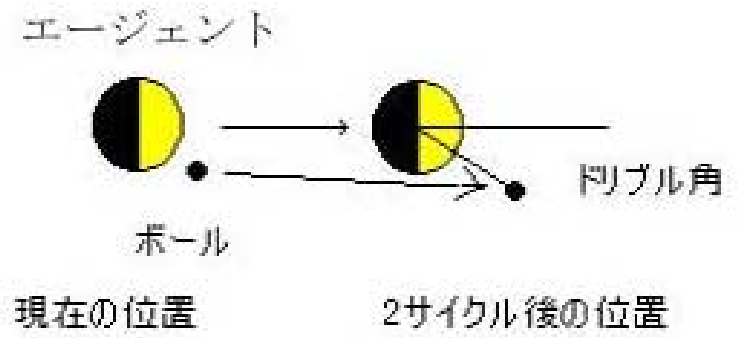


図 4.3 ドリブル角の図

4.3.2 スマートドリブル

スマートドリブルのアルゴリズムを書く上で必要なスキルが，回転キックである．回転キックとは，ボールを自分の周りでボールを回転させて勢いをつけ遠くに蹴る，RoboCup 特有のものである [3]．この回転キックは，RoboCup Japan Open などの大会において，上位チームは必ず取り入れている技術である．回転キックの動作を表した図 4.4 を示す．

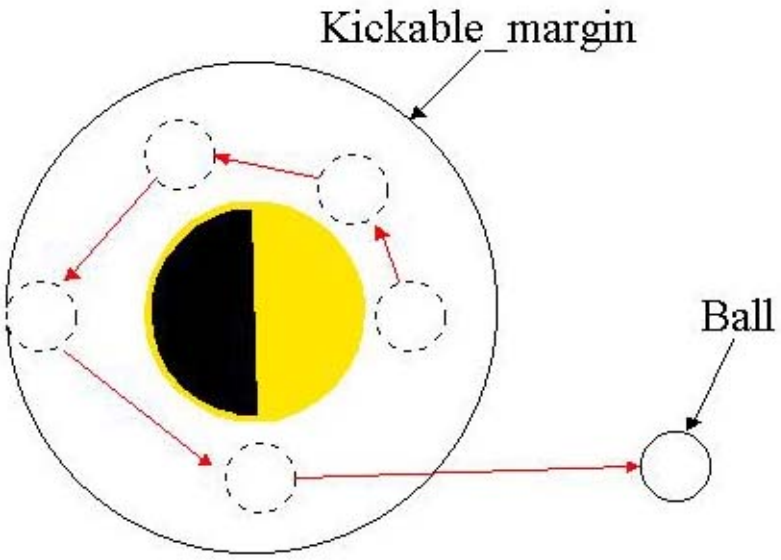


図 4.4 回転キック

ボールが自分の蹴ることの出来る範囲（ Kickable area ）内にあった場合に，ボールを蹴ることが出来る．回転キックは図 4.4 の様にボールを Kicable area 内で回し勢いをつける．ボールを Kickable area 内で蹴るためには，1 サイクル後の位置を指定して蹴る方向や強さを調節しなければならない．これを計算する方法を説明する．条件として，図 4.5 とする．

(p_x^t, p_y^t) から $(p^{+1}t_x, p^{+1}t_y)$ に，現在のボールの速度が (v_x^t, v_t^y) のボールを 1 サイクル後に移動させるために必要な蹴る強さ（ $Power$ ）と方向（ $Direction$ ）を求める．ここで，(a_x^t, a_y^t) はボールに加える加速度，θ_t はボールに加える加速度の絶対方向．θ_{t_kicker} は $Kicker$ の絶対方向をあらわしている．Soccer Server の物体の計算式から式をたてる [5].

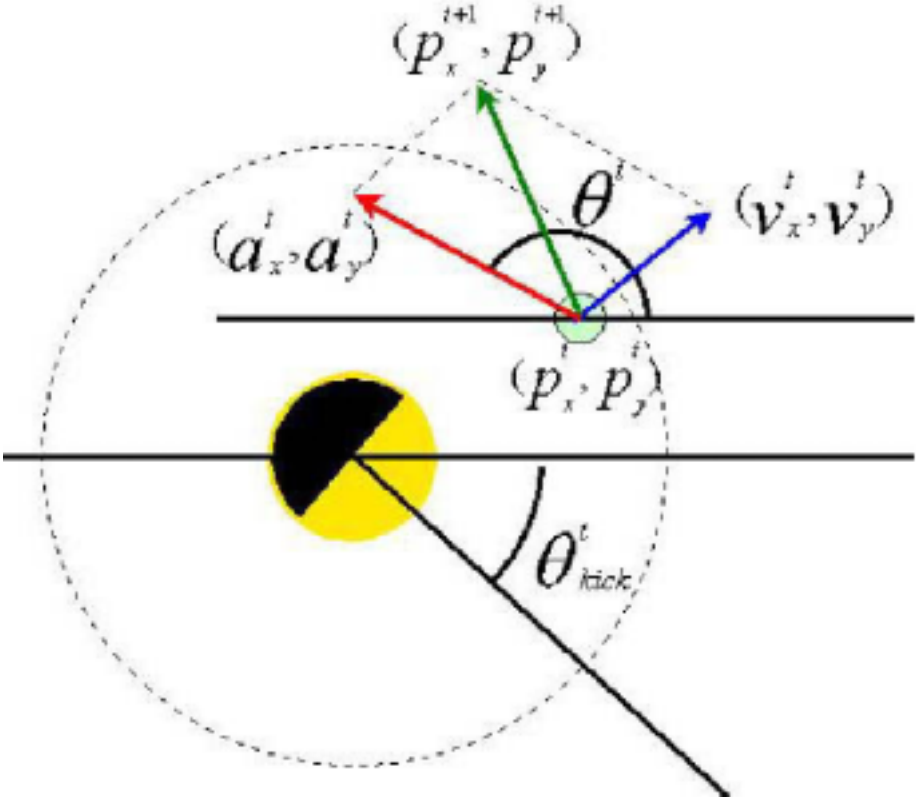


図 4.5　設定条件

$$(u_x^t, u_x^t) = (v_x^t, v_y^t) + (a_x^t, a_y^t) \tag{4.3}$$

$$(p_x^{t+1}, p_y^{t+1}) = (p_x^t, p_y^t) + (u_x^t, u_y^t) \tag{4.4}$$

上式から (a_x^t, a_y^t) を求める．

$$(a_x^t, a_y^t) = (p_x^{t+1}, p_y^{t+1}) - (p_x^t, p_y^t) - (v_x^t, v_y^t) \tag{4.5}$$

これにより，ボールに加えたい加速度が求まった．次に，絶対方向 θ_t を求める．

$$\theta_t = arctan(a_y^t / a_x^t) \tag{4.6}$$

次に，kick コマンドによって加わる加速度の計算式で求めた加速度 (a_x^t, a_y^t) をあらわすと

$$a_x^t = Power * Power\_rate * cos(\theta_t) \tag{4.7}$$

$$a_y^t = Power * Power\_rate * sin(\theta_t) \tag{4.8}$$

式 (4.7), (4.8) のあと $Power\_rate$ さえ求めれば $Power$ を求めることが出来る．$Power\_rate$ は，Soccerserver Manual[5] で定義されている式と定数を使い計算する．

最後に，$Direction$ を求める．サッカーサーバでボールに与える加速度の計算式は式 (4.9) の様になっている．

$$\theta_t = \theta_{t\_kicker} + Direction \tag{4.9}$$

この式から $Direction$ が求まる．

1 サイクル後の位置を指定して蹴ることが出来るようになれば，ボールの勢いが付くようにボールを蹴ればよい．

この回転キックを使ってドリブルをする事が，スマートドリブルの基本となる．スマートドリブルに回転キックが必要な理由は，スマートドリブルでは，敵エージェントが蹴れない範囲にボールを移動させることが目的となっている．そのために，回転キックのスキルを利用するのである．しかし，回転キックでは，回転する力を使って遠くに蹴ることが目的となっているために，そのままスマートドリブルに適用することは出来ない．

そのため，ボールを回すというアルゴリズムだけを適用する．

スマートドリブルのアルゴリズムは，distance と direction の二つから成り立っている．distance は，ボールを蹴る強さと同じ意味である．

$$distance1 = \sqrt{ball\_distance^2 - (ball\_size + player\_size)^2} \tag{4.10}$$

$$distance2 = \sqrt{(player\_size + (0.8 \times Kickable\_margin))^2 - x^2} \tag{4.11}$$

$$distance = distance1 + distance2 \tag{4.12}$$

これらの式 (4.10)，(4.11)，(4.12) を使用し，ボールを移動させる範囲を決定する．ball_distance とは，エージェントとボールとの相対的な距離で，そこから，ボールとエージェントの大きさを減算することによって，エージェントからボールまでの距離が distance1 で求まる．

相対距離とは，エージェントの中心から，指定した目標に対しての距離のことである．

distance2 では，Kickable_margin とボールとの距離を求めている．これらを加算することによって，ボールの移動できる距離が求まる．Kickable_margin とは，エージェントがボールを蹴ることの出来る範囲をいう．

direction は，以下の式 (4.13) を使用する．

$$direction = 180 + ball\_ang\_from\_body + (rotate) \times \theta_t \tag{4.13}$$

ball_ang_from_body とは，ボールと体の相対角度である．これを加えることによって，ボールがエージェントにぶつからないようにしている．

rotate は，敵エージェントのいない方向にボールを蹴るためのものである．

θ_t は，蹴りたい方向の絶対方向である．もしくは，自分がボールを任意の位置に移動させたいときの絶対方向である．

180 度を加えているのは，もし 30 度の方向に蹴ろうとした場合，30 度のままではそのまま 30 度の方向に蹴るだけで，敵エージェントに取られる可能性があるため，180 度を加えることによって，自エージェントを中心とした distance で求めた距離の円上を 30 度ずつ回転するようにし，ボールを敵エージェントに取られないようにしている．これを図 4.6 を示す．

相対角度とは，エージェントと指定された目標物に対しての角度のことをいう．指定した目標に対して，正面を 0 度とし，右側を正，左側を負とした角度を相対角度という．

絶対方向とは，エージェントを中心に敵ゴール側を 0 度とし，時計周りを正，反時計回りを負とする．範囲は 0 度 $\sim \pm$ 180 度表示で表現する．

これにより，ドリブル角と力を求めることが出来る．

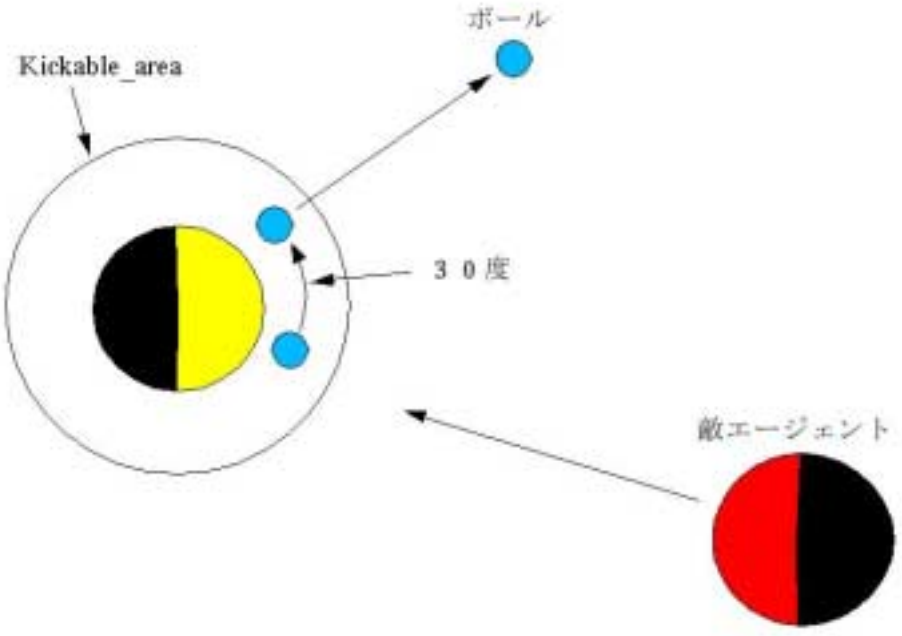


図 4.6　30 度回転させる場合

第 5 章

実験

5.1　実験方法

5.1.1　実験 1

実験 1 では，一体の敵エージェントをどのくらいの確率でかわせるのか，という実験を行った．

かわしたと判断されるのは，敵エージェントが自エージェントよりも後ろになった場合とした．これを図 5.1 に示す．

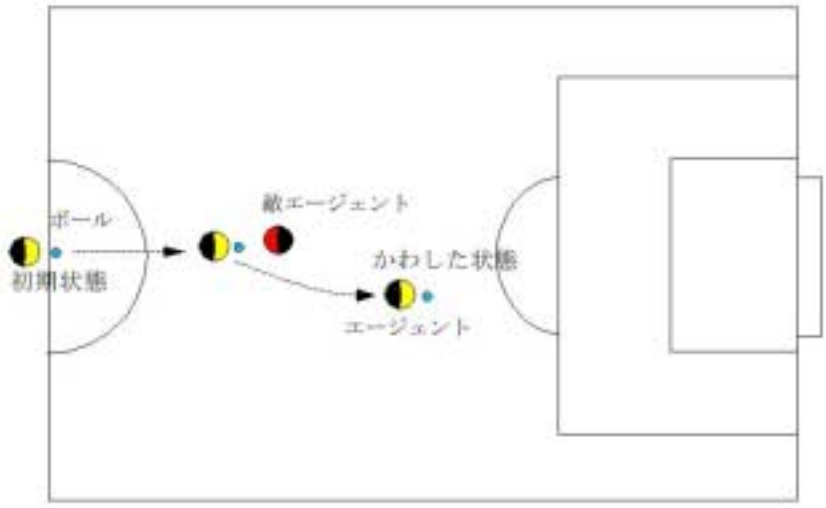


図 5.1　実験 1 の解説図

5.1.2　実験 2

実験 2 では，実験 1 と同じエージェントを使い，敵エージェント 2 体を使用してどれくらいの確率でかわせるか，という実験を行った．これを図 5.2 に示す．

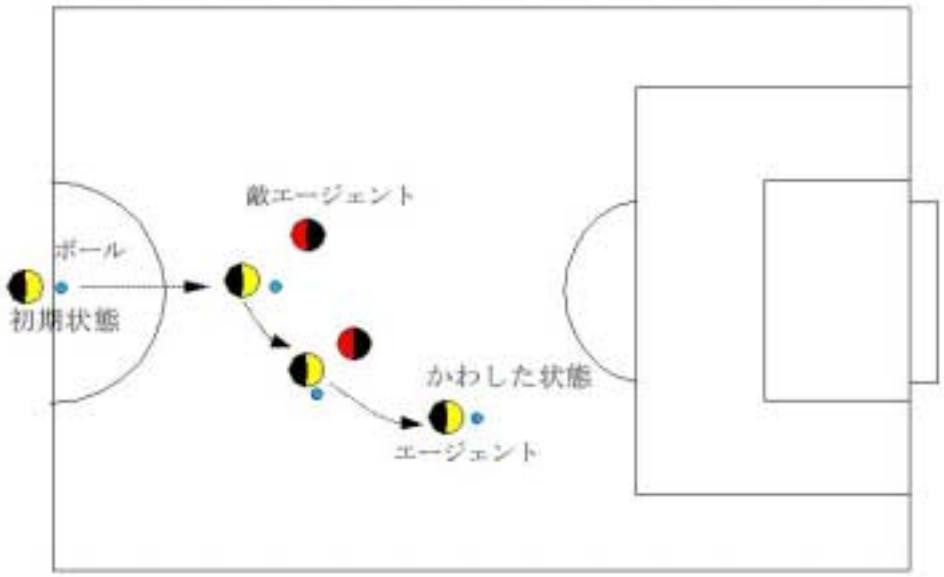


図 5.2　実験 2 の解説図

5.2　実験結果

5.2.1　実験 1 の結果

この実験では，敵エージェントの位置とボールの位置を確実に把握することが重要である．実験の結果，154 回行い 124 回成功したという結果となり，敵エージェントをかわす確率は，約 80 %となった．これを図 5.3 に示す．

その他は，スマートドリブルを使用しなかった場合である．実験結果は，スマートドリブルだけを使用して，敵エージェントをかわした場合の回数の確率を示している．

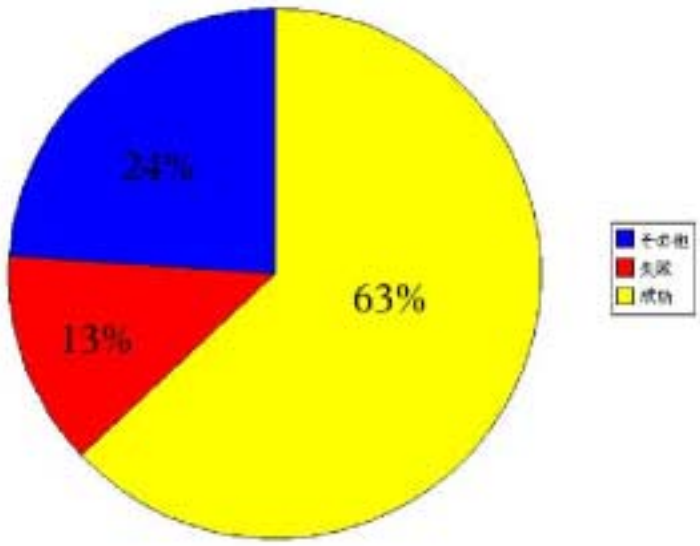


図 5.3 実験 1 の結果

5.2.2 実験 2 の結果

この実験では，敵エージェント二体の位置と自エージェントに対して，どちらが近くまたどちらがかわしやすいかということを，設定することが重要である．実験の結果，106 回行い 36 回成功，約 34 %となった．これを図 5.4 に示す.

その他は，スマートドリブルを使用しなかった場合である．実験結果は，スマートドリブルだけを使用して，敵エージェントをかわした場合の回数の確率を示している．

5.3 考察

5.3.1 実験 1 の考察

実験の結果だけを見れば，良好な結果だと思われる．敵エージェントも移動するので，かわす確率は，100 %という確率でかわすことは不可能だろうが，90 %ぐらいまでの確率にはもっていけるのではないかと推測される．そのためには，計算による誤差の修正が必要となってくる．

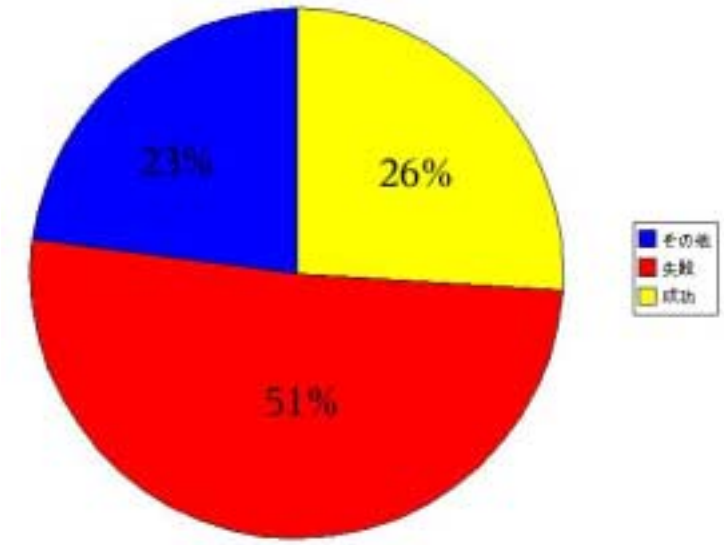


図 5.4 実験 2 の結果

また，改善すべき所はエージェントのボールの移動させる距離である．ボールを移動させる距離は，Kickable_area の大きさから，ボールの大きさを引いた距離ボールを移動させるので，敵エージェントとの距離が，サッカーサーバから得た情報よりも短い場合，ボールを移動させる距離が短いく，適切な方向でなければ，敵エージェントがボールを蹴れるためにボールを取られやすい，という場合があり，課題として残った．

そして，現段階ではボールを回している間は，エージェントは止まっているが，ボールを回している場合でも，移動をしているということが必要になるだろうと思われる．これを図 5.5 に示す．

こうすることによって，回転キックを使用せず，敵エージェントを避けることが出来るようになる．また，前述した課題にも対応出来るので，重要な課題として残った．

5.3.2 実験 2 の考察

実験 1 と比べて，かわす確率がかなり落ちたことが課題として残った．

- ボールを取られた状況

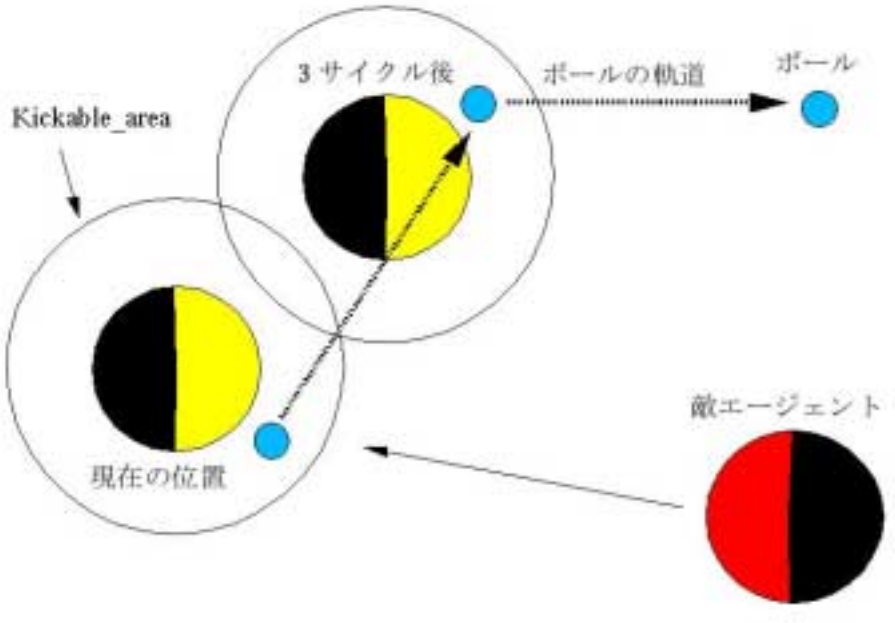


図 5.5 課題

主に，挟まれるように近付かれた場合で，近い方の敵エージェントに対してしか，避けようとしていなかったため，遠い方の敵エージェントに取られるというケースが目についた．これを図 5.6 に示す．

図 5.6 のような状況の場合，現状のスマートドリブルでは，かわせる可能性は，低いと思われる．そこで，早急にパスの実装が必要である．

- かわした状況

  主に，同じ方向から敵エージェントが近付いてくる場合に比較的にかわせていることが分かった．これを図 5.7 に示す．

  図 5.6 のような状況の場合，実験 1 とほぼ同じ状況となっているため，比較的にかわせていた．

  しかし，一体の敵エージェントをかわしている途中で，もう一体の敵エージェントにボールを取られる場合があったため，重み付けに対する課題が残った．

5.3　考察

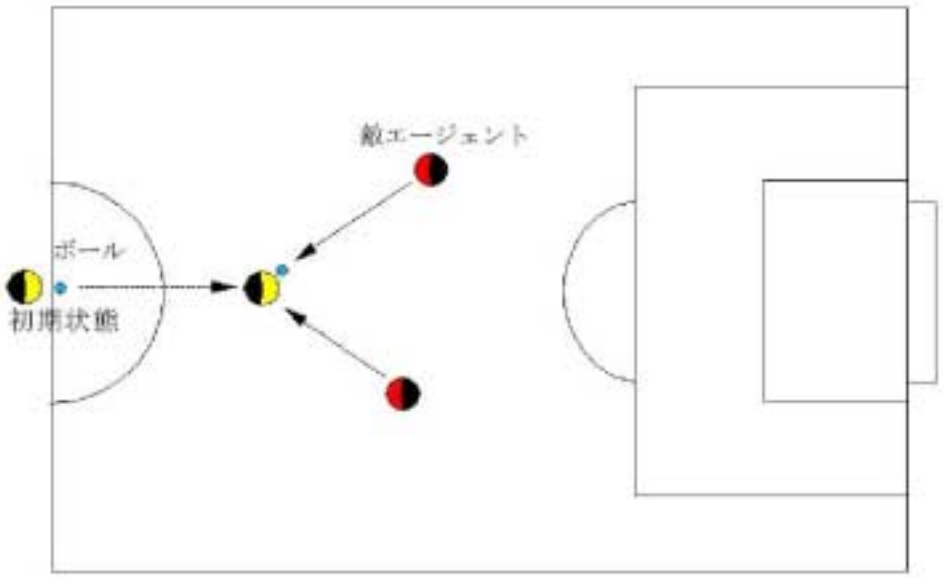


図 5.6　取られた場合

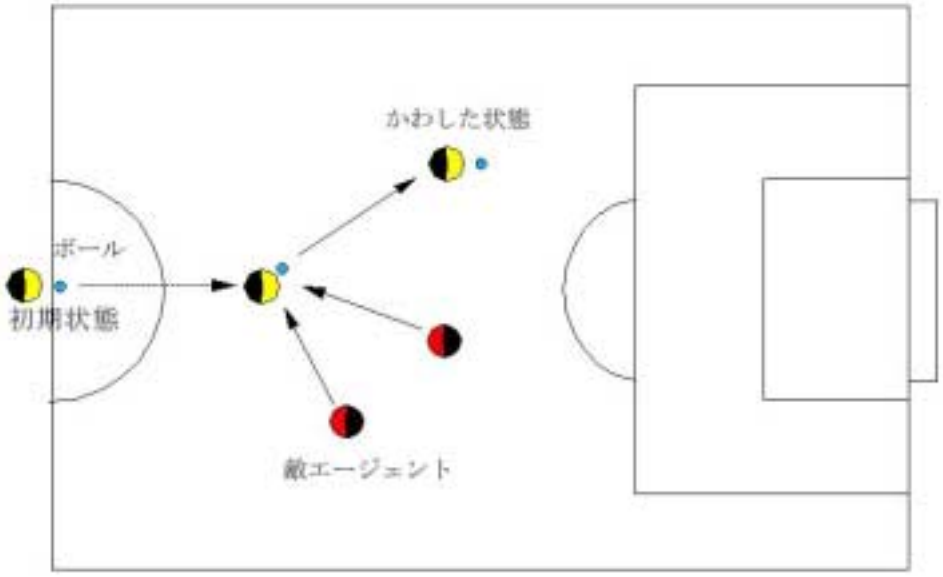


図 5.7　かわした場合

敵エージェントをかわせている場合というのは，敵エージェント一体の状況と似ているため，かわせていた場合がよくあった．

これらのことより，二体の敵エージェント近づいてくる場合，同じ方向からならば，敵エージェント一体と状況が似ているためかわせるが，挟まれるように近付かれた場合では，全ての敵エージェントの距離・角度などを，情報として把握していなければ，敵エージェントをかわすことは，ほぼ不可能であることが分かった．

また，ボールを取られた状況を分析し，ドリブル角に対する重み付けの検討が課題として残った．

第 6 章

まとめ

スマートドリブルの実装により，エージェントの性能強化に継った．これにより，エージェントの基本となるドリブルが出来るようになった．そして，クライアントの戦略・戦術部設計が行ない易くなった．

しかし，まだ計算による誤差の修正，重み付けの再検討の必要性等のために，完全なスマートドリブルには至っていないため，これらが今後の課題として残っている．このほかにも，近くにいる全ての敵エージェントの情報の把握も必要である．この点を，改良し来年度以降につなげていく必要がある．

また，パスの実装も課題としてあげられる．パスをする際には，ボールと味方エージェントの動きを予測しなければならない．そのために，補完・予測機能の強化が必要になって来る．予測の手法に時系列予測などの手法を使い，1 サイクル後の予測だけではなく，4 ，5 サイクル先の予測をする機能の追加も今後の課題の 1 つである．予測機能の強化により，より高度な戦略・戦術が設計できるようになる．

今後，KUT RoboCup Project と RAIK-NTG4 によって，代表チームが強化され RoboCup の大会での好成績が期待できる．

謝辞

本研究を進めるにあたりとても丁寧にご指導下さいました Ruck Thawonmas 助教授に心より御礼申し上げます.

本研究に関して色々な面で支えて下さった人工知能研究室の皆様に感謝します.

参考文献

[1] http://www.robocup.or.jp/

[2] http://sourceforge.net/projects/sserver/

[3] Peter Stone, "Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer", MIT Press, pp.241-243, 2000.

[4] http://ci.etl.go.jp/~noda/soccer/client/index.html.

[5] "Soccerserver Manual (日本語版)",
http://www.ita.tutkie.tut.ac.jp/~watta/RoboCup/jmanual.html.

[6] Ruck Thawonmas, 大森洋一, 平山純一郎, "RoboCup ソフトウェアロボットづくりによる問題設計解決型学習", 第 9 回情報教育方法研究発表予稿集, pp.50-51, July. 2001.