解説

RoboCup サッカー 3D シミュレーションリーグ 解説：競技会の概要とエージェント開発†

中島 智晴*・中邨 陽介*・炭谷 剛志*

1. はじめに

RoboCup サッカー 3D シミュレーションリーグのチーム開発を行うためには、ヒューマノイドロボットの関節の動きを設計する必要がある。また、ロボット工学や運動力学に関する専門知識が要求され、人工知能を専門とする開発者には敷居が高いことがある。そのような場合、低水準な動きについては他チームが公開しているソースコードをベースにすることで、高レベル意思決定、特に学習システムに関する部分だけに集中して開発が可能である。前稿では、3D リーグのシミュレーションの仕組みと環境構築に関して説明した。本稿では、前稿の復習を交えながら 3D シミュレーションリーグの競技会およびエージェント開発の変遷について説明し、ベースコードからのエージェント開発の手順を具体例を挙げて解説する。サンプルチームとして、2010年に公開されたチーム opuCI 3D のソースコードを使用する。国内大会に参加するのに十分な 3D エージェントを作成できるように詳述したので、大会参加に興味のある読者は活用していただきたい。

2. 3D シミュレーションの競技会

RoboCup サッカー 3D シミュレーションリーグは、2004年から正式競技として導入された。2007年以降はブレイヤエージェントとしてヒューマノイドロボットを用いた競技が行われている。表 1 に、ヒューマノイドロボットが採用されてからの国際大会の概要を示す。参加チーム数は、予備審査を経て国際大会に参加することができたチーム数であり、予備審査登録 (Preregistration) したチーム数はこれよりはるかに多い。はじめて採用されたヒューマノイドロボットは HOAP-2 というロボットをモデルに製作されたブレイ

ヤで、soccerbot と呼ばれる (図 1)。2008年から現在にかけては、Aldebaran 社の開発した Nao というロボットをモデル化したブレイヤ (図 2) が使用されている。Standard Platform リーグでは、実機の Nao ロボットを用いた試合が行われており、同リーグと連携することで、3D リーグで得られた知見を実機に転用することが期待される。

表 1 から分かるとおり、チームを構成するエージェントの数は 2, 3, 4, 6 と年々増加している。エージェントの増加に伴って、試合中にエージェント同士が衝突した時、物理計算の複雑さからシミュレーションに大きな負荷がかかることが頻繁に発生するようになった。多数のエージェントが複雑にもつれ合ってしまうと負荷は更に強くなり、シミュレーション速度が大幅に低下し、場合によってはシミュレータが暴走もしくは停止してしまうこともあった。そのため、競技には人間の審判を用意し、エージェントの衝突のためシミュレータに過度の負荷が掛かりそうな状況になった場合は手作業でエージェントを移動させ、仕切り直しにするという手段が取られていた。また、2010年から、ボール周辺には両チームとも 1 体のエージェントしか存在できない、というルールが定められた。このルールを破ってボールに接近したエージェントはサーバに組み込まれたレフェリーによってフィールドの外側に強制移動させられる。上記のルールの影響で、エージェントの激しい衝突による遅延は解消されたが、状況によっては理不尽な強制移動が発生する (例えば、複数のエージェントがゴール前でディフェンスしようとしても、1 体になるまで強制移動させられるなど)。そこで、2011年の世界大会ではさらにルール変更があり、3 体以上のエージェントが接触し

表 1 国際大会 (ヒューマノイド採用後) の開催地と概要

年度	開催地	ロボットモデル	ブレイヤ数	参加チーム数
2007	アトランタ	HOAP-2	2	32
2008	蘇州	Nao	3	23
2009	グラーツ	Nao	4	28
2010	シンガポール	Nao	6	30

† RoboCup Soccer 3D Simulation League : Overview of Competitions and Agent Development
Tomoharu NAKASHIMA, Yosuke NAKAMURA and Takeshi SUMITANI

* 大阪府立大学大学院
Osaka Prefecture University

図1 soccerbot

図2 Naoモデル

た場合、最後に接触したエージェントが強制移動させられるようになった。しかし、問題の完全解決には未だ至っておらず、より公正なルールの整備が待たれる。

3. エージェント開発

3.1 概要

現在プレイヤエージェントとして採用されているNaoモデル(図2)には22箇所の関節があり、エージェントは、作動させる関節とそのトルクをサーバに送信する。そのため、エージェント開発では、パス、ドリブルなどといったサッカーを行う上で要求される行動をロボットの関節角度の動きに置き換える仕組みを持つ必要がある。

3Dリーグのエージェント開発は高レベル行動、低レベル行動の二つを考慮する必要がある(図3)。高レベル行動とは、1体もしくは複数のエージェントを伴う戦術や戦略などに深く関わる行動である。具体的には、パス、ドリブルなどの行動選択や目標設定、フォーメーションなどが含まれる。低レベル行動は、高レベル行動を実装するための基礎となる行動である。例えば、歩行動作やキック、起き上がり動作など

図3 エージェント開発における階層構造

が挙げられる。高レベル行動と低レベル行動はお互いに依存するものであり、チーム開発の際には、これら両方の動作の完成度を同時に上げていく必要がある。シミュレーションリーグは本来、実機の構造に依存する部分を排除し、高レベル行動の開発に重点を置くことを目的として設立されたものである。実際、戦術、戦略の学習を目的としてRoboCupシミュレーションリーグに参入しようとしている方もいるはずである。しかし、低レベル行動を実装しないことには高レベル行動の開発は不可能であるため、すべてのチームはまず低レベル行動を用意することから開発を始めることになる。低レベル行動の開発には、動作の要となるポーズの関節角度をキーフレームとして保存しておき、キーフレーム間の関節角度は線形補間などにより求める手法[1]や、周期関数によって関節角度の目標値を記述する手法[2, 3]や、理想的な手足先端部の位置から逆キネマティクスにより目標角度を求める方法[4]もある。4章以降では、こういった低レベル行動があらかじめ作られているベースエージェントの利用について説明を行う。

3.2 ロボットモデルの変遷と開発のトレンド

最初のヒューマノイドモデルとして採用されたsoccerbot(図1)は、身長およそ5メートルに対して体重がおよそ5kgとアンバランスな設定であったため、相手エージェントや障害物との衝突で大きく飛び跳ねたり勢い良く転倒することが多かった。関節は各腕に

4 箇所、各足に 6 箇所の合計 20 箇所存在し、角度設定には制限がほとんどなかった。そのため、例えば仰向け、うつ伏せのどちらの状態からでも全く同じ動作で起き上がることができるなど、人間には到底不可能な挙動が可能であった。また、上部部の質量比が高いため、重心が高く転倒しやすかったが、一方で関節を動作させるためのモーターに強力なものが設定されていたためキック力は強く、フィールド中央からキックして、相手のキーパを吹き飛ばしつつゴールするような強烈なシュートを打つことが可能であった。当時の競技会(図 4)では、ほとんどのチームが低レベル行動の実装に苦勞しており、何とか歩行動作が行えるが、頻繁に転倒してしまうといった様子であった。

2008 年から soccerbot に代わって採用された Nao モデル(図 2)は身長がおよそ 0.57 メートル、体重 4.5kg と小型で、モーターの出力もかなり小さく設定されている。そのため、衝突時に飛び跳ねることはなくなったが、キックの力も弱まってしまい、ボールを浮かせるようなシュートを打つことが非常に難しくなった。また、関節は実機の Nao ロボットになるべく近くなるように設計されており、駆動角度に制限があるなど soccerbot とは大きく異なっている。そのため以前の

図 4 2007 年決勝戦の様子(soccerbot エージェント)

図 5 2010 年決勝戦の様子

リソースはほとんど再利用できなくなり、開発者は改めて低レベル行動からの実装を強いられることとなった。また、Nao モデルの公開は国際大会のわずか 2 ヶ月前であったため、十分な開発時間が取れず、歩行することで精一杯なチームも多かった。そのような状況下で、2008 年度優勝チームの SEU-RedSun は、競技中にエージェントを 1 体しか接続できず、当然キーパもない単独エージェント状態で 3 体の敵エージェントを相手に勝利するなど、優れたパフォーマンスを見せた。これはつまり、低レベル行動の完成度が高ければそれだけで勝利できていたということである。やがて年が経つごとに低レベル行動の完成度は上がっていき、最近では多くのチームが優れた低レベル行動を実装している。それに合わせて視界制限の導入やフィールドサイズの拡大など、競技設定も変更されていった。2010 年の競技会(図 5)では、味方にパスを出すような戦略行動を実装したチームも見られた。

3.3 ベースコード

rcssserver3D パッケージ[5]には、サーバとの接続やサーバから受け取った情報の解釈などに利用できるライブラリや、それらを利用したサンプルエージェントである rcssagent3d が用意されている。開発者は、rcssagent3d のソースコードを基にしてチーム開発を行うことができる。しかし、rcssagent3d には、歩行動作、キック、起き上がり動作といった低レベル行動は一切用意されておらず、関節制御の仕組みも十分に整っているとは言えない。そのため、rcssagent3d をベースにチーム開発を始める場合は、まず関節制御の仕組みから作る必要がある。

一方で、競技会に参加しているチームの中には、ソースコードを公開しているチームがある。戦術部分に焦点を当てて開発したいチームは、サンプルエージェントから作成するよりも、公開されているチームのソースコードを基に開発したほうが効率が良く便利である。ただし、ベースコードが著作権(例えば、GPL)を主張している場合は、手を加えたソースコードやバイナリの公開に際して、ライセンスに準拠した対応が必要である。また、ソースコードごとに異なる特徴があるので、様々なチームのソースコードを試して、相性の良いベースチームを使って開発を進めるとよい。以下に、公開されているソースコードのうち、代表的なものを紹介する。

- littlegreenbats

このチームは 2007 年の世界大会で 2 位となったが、この時使用されたエージェントのソースを公

開している。また、2009年には、エージェント開発に有用な関数を作成して取りまとめたライブラリlibbatsのソースを公開している[6]。

- **nexus**

ベースコードとして利用されることを想定した、視界制限が導入されたエージェントに必要な座標計算などが実装されたチームを公開している[7]。

- **FUT-K**

福井工業大学のチーム。2009年に歩行、起き上がり動作が実装されたエージェントを公開している。また、2010年には同チームが国際大会で使用したキーパのコードを公開している[8]。

- **opuCI_3D**

大阪府立大学のチーム(2008年はOPU_hana_3D)。世界大会で使用したチームのソースコードを公開している[9, 10]。2010年には、チームのエージェントが使用している基本スキルを全て利用できるソースコードを公開している[11]。

4. opuCI_3D

4.1 概要

本章では、2010年度RoboCup秋キャンプ[11]で配布されたopuCI_3Dのソースコード(opuCI_3D_camp)を利用したエージェント開発の実例を示す。opuCI_3D_campは、チームlittegreenbatsによって開発されたライブラリであるlibbatsのサンプルエージェントに、opuCI_3Dの2010年度の基本動作と、簡単な行動ルールを追加したものである。libbatsはGPLに基づいたフリーソフトなので、opuCI_3Dを修正した場合は、そのソースコードの公開もGPLの取り決めに準拠する必要がある。opuCI_3D_campには単純な挙動のアタッカ、ディフェンダ、キーパがあらかじめ用意されており、実装されている動作を組み合わせることで利用できる枠組みが備わっている。opuCI_3D_campは、2体のアタッカ、3体のディフェンダ、1体のキーパで構成される。各エージェントは、初期状態ではそれぞれ次のような行動を取る。

- **アタッカ**

背番号2, 3のエージェントが担当する。2体のうち、ボールに近い側のエージェントがメインアタッカ、遠い側のエージェントがサブアタッカとなる。この役割は、試合中に状況に応じて切り替わる。メインアタッカは、相手ゴールの中心に向かってドリブルし、得点を狙う。サブアタッカは、ボールから味方ゴール側に2メートル移動し

た地点に移動し、待機する。

- **ディフェンダ**

背番号4, 5, 6のエージェントが担当する。基本的にそれぞれ指定されたポジションで待機する。ボールが自分自身の周辺1.5メートル以内に存在する場合、ボールに向かって移動し、キックする。

- **キーパ**

キーパ背番号1のエージェントが担当する。味方ゴール中心からボール側に1メートル移動した地点で、ボール側を向いて待機する。

4.2 実行

opuCI_3D_campのコンパイルは次の手順で行う。

1. 必要なパッケージのインストール
2. ソースコードのダウンロード
3. libbats-2.0_opuciのインストール
4. opuCI_3D_campのコンパイル

なお、本稿を執筆時点で著者はUbuntu10.10がインストールされた計算機上で確認を行っている。また、3Dシミュレーションの開発環境[5]は前稿の解説に従って既にインストールされているものとする。まず、エージェントおよびライブラリのコンパイルに必要なパッケージのインストールを行う。以下に必要なパッケージをインストールするコマンドを示す。なお、サーバインストールに必要なパッケージは前稿でインストール済と仮定しているため省略している。

```
$ sudo apt-get install libxml2-dev autoconf
libsigc++-dev libeigen2-dev ccbuild doxygen
```

次に、ソースコードを入手する。ソースコードは、国内のRoboCup競技会などの情報が纏められているWebページであるSimJP[11]からダウンロードできる。エージェントを構成するファイルを以下に示す。これらをSimJPからダウンロードする。

- libbats-2.0_opuci.tar.gz
- opuCI_3D_camp.tar.gz

次に、libbats-2.0_opuciのインストールを行う。これは、littegreenbatsが公開しているソースコードの一部に変更、追加を加えたものである。コマンドラインでlibbats-2.0_opuci.tar.gzをダウンロードしたディレクトリに移動して、以下のコマンド操作を行うことでインストールできる。


```
$ tar zxvf libbats-2.0_opuci.tar.gz
$ cd libbats-2.0_opuci
$ ./configure
$ make
$ sudo make install
```

また, libbatsにはdoxygenにより生成することができるドキュメントが用意されている. このドキュメントを参照することで, libbatsに含まれるクラスや関数を調べることができる. ドキュメントの有無は開発効率に大きく影響を与えるため, 是非利用したい. ドキュメントの生成は以下のコマンドで行う.

```
$ make doc
```

これにより, docsディレクトリ内にドキュメントが生成される. htmlディレクトリ内のindex.htmlをブラウザで開くことで, ドキュメントが参照できる. 最後に, opuCI_3D_campのコンパイルを行う. libbatsと同様に, ダウンロード先のディレクトリで以下のコマンド操作によりコンパイルする.

```
$ tar zxvf opuCI_3D_camp.tar.gz
$ cd opuCI_3D_camp
$ ./bootstrap.sh
$ ./configure
$ make
```

rcsoccersim3dコマンドでサーバを起動したあと, 実行スクリプト(start.sh)を実行する. モニタに6体のエージェントが表示されていれば成功である.

4.3 修正の実例

前節でインストールしたopuCI_3D_campには基本的な動作のみが実装されているため, 競技会参加や研究の題材として使用するには, 必要に応じて修正しなければならない. 本節では, opuCI_3D_campに対して, 次の2つの変更を適用する手順を説明する.

- 行動内容の修正: アタッカが相手ゴール周辺でシュートするように変更する.
- プレイヤ数の修正: アタッカ, ディフェンダを追加し, エージェント9体で構成されるチームに変更する.

4.3.1 行動内容の修正

エージェントの行動は, xmlファイルに記述された木構造にしたがっている. 図6に, opuCI_3D_campのアタッカの行動木の一部を示す. また, 図7に,

図6 アタッカの行動木の一部

図7 xmlファイルの構成

xmlファイルの構成を示す. 図6において, 各ノードは行動を示す. 行動木において, 同レベルの行動の優先順位は, 行動ごとに定義された信頼度に応じて決定する. 例えば, 初期設定では, KickTo行動の信頼度が0.6, DribbleTo行動の信頼度が0.7で固定されているため, メインアタッカはAttack実行時には常にDribbleToを実行する. 試合状況に応じて信頼度の値を動的に変更することで, 行動の切り替えを行うことができる. 信頼度は, 各行動クラスのgetCapability関数および各行動を記述しているxmlファイル内で決定されている. よって, ゴール周辺でキック, それ以外でドリブルを行うためには, ゴール周辺でのみキックの信頼度を高めるように設定すればよい.

リスト1に, confinc/commonbehavior.xmlの175行目に記述されているKickTo行動の記述例を示す. リスト1において, capabilityタグが信頼度の値を示す. xmlファイルでは, low, highの2通りの信頼度を設定することができる. また, リスト2にKickTo行動のgetCapability関数(Behavior/KickTo/getCapability.cc)の記述例を示す. リスト2において, この例でのgetCapability関数では, ボールと相手ゴールのローカル座標を取得し, ボールから相手ゴールへのベクトルの長さ(つまり, ここではボールからゴール中央までの距離)に基づいて戻り値(信頼度)を設定している.

リスト 1：行動の記述例

```
<behavior type="KickTo" id="kickto">
  <param>
    <capability low="0.0" high="0.6" />
  </param>
  <slot index="0-0">
    <behavior refid="positioning"/>
  </slot>
  <slot index="1-0">
    <behavior refid="turnaround"/>
  </slot>
  <slot index="2-0">
    <behavior refid="gotokickpos"/>
  </slot>
  <slot index="3-0">
    <behavior refid="swingrightleg"/>
    <behavior refid="swingleftleg"/>
  </slot>
</behavior>
```

リスト 2：getCapabilityの記述例

```
ConfidenceInterval KickTo::getCapability()
{
  Localizer& loc = SLocalizer::getInstance();

  Vector3d ball =
    loc.getLocationLocal(Types::BALL)->getMu();
  Vector3d theirGoal =
    (loc.getLocationLocal(Types::GOAL1THEM)->getMu()
    + loc.getLocationLocal(Types::GOAL2THEM)->getMu())
    / 2.0;
  Vector3d ballToTheirGoal = theirGoal - ball;

  return capable(ballToTheirGoal.norm() < 3.0);
}
```

capable関数は、引数が真であればxmlファイルのhighに設定された信頼度を返し、偽であればlowに設定された信頼度を返す。したがって、この例では、ボールからゴール中央までの距離が3.0より小さければhigh(0.8)を、そうでなければlow(0.0)を返す。

以上まとめると、ゴール周辺でドリブルからキックに切り替えるよう修正するには、confinc/commonbehavior.xmlに記述されているKickTo行動のcapabilityパラメータhighの値をDribbleToより高くした上で、KickToのgetCapabilityの戻り値であるcapable関数の引数に、相手ゴール周辺にボールが存在すれば真、そうでなければ偽となるbool値を指定すればよい。すなわち、KickTo行動のgetCapability関数をリスト2のように変更すればよい。

4.3.2 プレイヤ数の修正

3Dシミュレーションリーグの2011年度の公式大会では、1チーム当たりのプレイヤ数を9体にすることが計画されている(2011年3月現在)。そのため、opuCI_3D_campを利用して公式大会に参加するためには、プレイヤ数を9体に設定し直さなければならない。本節では、opuCI_3D_campのプレイヤ数の設定およびプレイヤの役割の設定方法を説明する。

プレイヤの役割は、opuCI_3D_conf.xml内で背番号ごとに記述されている。初期状態では、xmlファイル内では背番号7以降のプレイヤは全てattackerに設定されているので、例えば背番号7をattacker、背番号8と9をdefenderとして追加する場合は、背番号8と9の役割をdefenderに変更すればよい。なお、新たに設定されたアタッカとディフェンダのポジションは、個別に指定する必要がある。アタッカが3体以上の場合は、ボールに最も近い1体がメインアタッカとなり、それ以外の全アタッカがサブアタッカとなる。

また、アタッカが3体以上の場合、サブアタッカを担当するエージェントが複数存在することになるが、初期設定ではこれらのエージェントの待機地点が全て同じになり、待機地点周辺で衝突が発生してしまう。これを回避するために、個別に待機地点を設定する。アタッカのポジションはSupportAttack行動のgenerateGoal関数(Behavior/SupportAttack/generateGoal.cc)で設定する。リスト3にgenerateGoal関数の編集例を示す。簡単のため、一部省略して記述している。

ディフェンダのポジションはdefender_pc.xmlの70行目に記載されているwhereタグのパラメータを編集することで決定する。はじめから設定されているディフェンダと同様に、新たに追加された背番号8、9のポジションを設定すればよい。最後に、実行スクリプトであるstart.shの9行目に記述されているプレイヤ数を6から9に変更することで、9体のエージェントが起動できるようになる。

リスト 3：generateGoalの編集例

```
rf<Goal> SupportAttack::generateGoal()
{
  // goal, dis, con, loc の宣言および値の代入

  AgentModel\& am = SAgentModel::getInstance();
  Vector3d ball =
    loc.getLocationLocal(Types::BALL)->getMu();
  Vector3d ourGoal =
    (loc.getLocationLocal(Types::GOAL1US)->getMu()
    + loc.getLocationLocal(Types::GOAL2US)->getMu())
    / 2.0;

  Vector3d pos2 =
    ball + (ourGoal - ball).normalized();
  Vector3d pos3 =
    ball + (ourGoal - ball).normalized() * 2.0;
  Vector3d pos7 =
    ball + (ourGoal - ball).normalized() * 3.0;

  // angle の宣言および値の代入

  if( am->getNum() == 2 )
    con->addVar("Pos", pos2); // 背番号2 の待機地点
  if( am->getNum() == 3 )
    con->addVar("Pos", pos3); // 背番号3 の待機地点
  if( am->getNum() == 7 )
    con->addVar("Pos", pos7); // 背番号7 の待機地点

  con->addVar("Angle", angle, angle);

  return goal;
}
```

4.3.3 競技会参加に向けて

競技会では、自チームの成果を確認できるだけでなく、他の開発者との意見交換ができ、効果的なチーム改良の参考になる。他チームの動作を見て参考になる事も多い。チーム開発の成果を試すにはこれ以上の機会はないだろう。ひと通り動作するエージェントが完成したら、是非RoboCup競技会への参加を検討していただきたい。また、本稿で紹介したエージェントのさらなる拡張例としては、

- ディフェンダ、キーパの行動作成
- フォーマーションの設計
- 低レベル行動の追加
- キックイン、ゴールキック、コーナーキックの実装

などが考えられる。

国内大会(RoboCup Japan Open)に参加するには、まず1月頃に仮登録を、3月上旬に本登録を行う。詳細はSimJP[12]のJapanOpen関連ページを参照するとよい。世界大会の場合は、2月頃に仮登録を行い、チームの特徴をまとめたTeam Description Paper (TDP)、昨年度の優れたチームとの試合のログファイル、過去のRoboCup Symposiumでの業績などをまとめたQualification Materialを提出する。その後、運営側によって審査が行われ、参加チームが決定される。なお、Qualification Materialに必要なものは年によって異なり、詳細は3Dのメーリングリストでアナウンスされる。

5. おわりに

本稿では、3Dシミュレーションのエージェント開発について説明した。また、チームopuCL3Dの公開コードを利用したエージェント開発の実例について解説した。3Dシミュレーションリーグは、立ち上げ当初はシミュレーション形態やエージェントモデルが定まらず、安定した競技が行えていなかったが、2008年にエージェントがNaoモデルに固定化されてからは着実に進歩してきており、最近の国際大会では洗練された低レベル行動が実装されたチームによる高度な試合が展開されるようになっている。一方で、日本国内の競技会のレベルは国際大会と比較してやや劣っており、参加チーム数も例年10チーム弱と少数である。本論文が、国内の3Dリーグ参加者を増やすことに貢献できれば幸いである。

6. 本稿に関連する便利な URL

以下に、3Dリーグのエージェント開発を行う上で役に立つと思われるURLを示す。参考文献と併せて、エージェント開発時に利用していただきたい。

● SimJP

主に国内の競技会の情報が公開されるページ。
<http://rc-oz.sourceforge.jp/pukiwiki/>

● サッカーシミュレーション3D体験LiveDVD

3D環境を体験してもらうためのLiveDVD。インストール作業不要。以下のページからダウンロード可能。

<http://rc-oz.sourceforge.jp/pukiwiki/index.php?Software%2FLiveDVD>

● 3Dサーバダウンロードページ

3Dサーバをダウンロードできるページ。
<http://sourceforge.net/projects/simspark/files/>

● SimSpark wiki

SimSparkやrcssserver3Dのインストール手順や仕様が記載されているページ。

<http://simspark.sourceforge.net/wiki/index.php/MainPage>

● メーリングリスト[Sserver-three-d]

3Dシミュレーション全体のメーリングリスト。国際大会のルールや参加に関する情報は主にこのメーリングリストで公開される。

<https://lists.sourceforge.net/lists/listinfo/sserver-three-d>

● メーリングリスト[robocup-sim]

サッカーシミュレーションリーグ全体のメーリングリスト。

<https://lists.cc.gatech.edu/mailman/listinfo/robocup-sim>

● メーリングリスト[rc-simjp]

国内のシミュレーションリーグ関連のメーリングリスト。

<http://lists.sourceforge.jp/mailman/listinfo/rc-oz-simjp>

● YouTube動画

数多くの3D競技を収めた動画がある。

<http://www.youtube.com>

検索例：RoboCup 3D simulation

参 考 文 献

- [1] C. Bustamante, X. Pérez, N. Liarte, C. Flores, and G. Leonardo, "Borregos3D Team Description 2008," RoboCup 2008 Team Description Papers, CD-ROM, 5p, 2008.
- [2] N. Shafii, O. M. Nezami, S. Aslaniand S. S. Ghidary, "Evolution of Biped Walking using Truncated Fourier Series and Particle Swarm Optimization," In RoboCup 2009 : Robot Soccer World Cup XIII, pp.344-354, 2009.
- [3] N. Shafii, L. P. Reisand N. Lau, "Biped Walking using Coronal and Sagittal Movements based on Truncated Fourier Series," In Robocup2010 : Robot Soccer World Cup XIV, pp.79-92, 2010.
- [4] J. Maand S. Cameron, "OxBlue2008 (3D) Team Description 2008," RoboCup2008 Team Description Papers, CD-ROM, 7p., 2008.
- [5] Spark-A generic physical simulator, Sourceforge プロジェクト, <http://sourceforge.net/projects/simspark/files>
- [6] The Little Green BATS, Web ページ, <http://www.littlegreenbats.nl>
- [7] Nexus Robocup Soccer Simulation Team Project, Web ページ, <https://launchpad.net/robonexus>
- [8] SimJP, 3D/ 公開ソースリスト FUT-K, <http://rc-oz.sourceforge.jp/pukiwiki/index.php?3D%2F%B8%F8%B3%AB%A5%BD%A1%BC%A5%B9%A5%EA%A5%B9%A5%C8>
- [9] OPU_hana_3D, 公開ソース, <http://www.cs.osakafu-u.ac.jp/~nakashi/download/hana083Dsrc.tar.gz>
- [10] opuCI_3D, 公開ソース, <http://www.cs.osakafu-u.ac.jp/~nakashi/download/opuCI3DRC2010.tar.gz>
- [11] SimJP, 2010 年度秋キャンプについて, <http://rc-oz.sourceforge.jp/pukiwiki/index.php?Event/2010/Camp>
- [12] SimJP, Web ページ, <http://rc-oz.sourceforge.jp/pukiwiki/>
(2011年7月15日 受付)

[問い合わせ先]

〒599-8531 大阪府堺市中区学園町 1-1
大阪府立大学大学院 工学研究科 電気・情報系専攻
中島 智晴
TEL : 072-254-9351
FAX : 072-254-9915
E-mail : nakashi@cs.osakafu-u.ac.jp

著 者 紹 介

なかしま ともはる
中島 智晴 [正会員]

2000 年大阪府立大学大学院工学研究科電気・情報系専攻博士後期課程修了。同年同大学大学院工学研究科助手, 2001 年同講師, 2005 年同助教授, 2007 年同准教授, 現在に至る。博士(工学)。計算知能の基礎研究とその学習エージェント, データマイニング, ロボカップサッカーへの応用に関する研究に従事。IEEE, 計測自動制御学会, 日本知能情報ファジィ学会, 日本経営工学会の会員。

なかむら ようすけ
中村 陽介 [非会員]

2009 年大阪府立大学工学部知能情報工学科卒業。2011 年同大学大学院工学研究科電気・情報系専攻知能情報工学分野修了。2011 年 4 月よりシャープ株式会社。在学中は RoboCup サッカーシミュレーション, 強化学習の研究に従事。

すみだに たけし
炭谷 剛志 [非会員]

2010 年 4 月研究室配属。2011 年大阪府立大学工学部知能情報工学科卒業予定。ロボカップサッカーシミュレーション 3D のチーム開発担当。