CONSTRUCCIÓN DE AGENTES EN EL SIMULADOR DE LA ROBOCUP MEDIANTE LA INTERFAZ ATAN

Construction of agents in the RoboCup Simulator using the interface ATAN

M.Sc. Yohoni Cuenca S.
yohoni@gmail.com

RESUMEN

En el presente trabajo se describe la construcción de agentes mediante la interfaz Atan, así se analizan bastantes características del simulador 2D de la RoboCup descrito en Noda et al (1997), Chen et al (2003) y Kitano et al (1997), como ser los sensores y efectores de los jugadores, los objetos del campo de juego y el ruido. Para lo cual se implementan programas en java mediante la interfaz Atan desarrollada por Wagner (2002), para analizar todos estos aspectos. Los programas implementados muestran aspectos básicos como la construcción de comportamientos, la construcción de equipos y la conexión del equipo al servidor, así también se hace una descripción del campo respecto a sus dimensiones y su representación en el plano cartesiano.

Palabras clave:

Futbol robótico, Servidor de la RoboCup, sistemas multi-robot, Interface ATAN, JAVA.

ABSTRACT

In this paper we describe the construction of agents by Atan interface and many features are analyzed from the RoboCup 2D Simulator described in Noda et al (1997), Chen et al (2003) and Kitano et al (1997), as be the sensors and effectors of the players, the playing field objects and noise. For programs which are implemented in java by Atan interface developed by Wagner (2002) to analyze all these aspects. Implemented programs show basic aspects of behavior such as behaviors building, team building and connecting the team to the server, and also provides a description of the field relative to its size and its representation in the Cartesian plane.

keywords:

Robotic soccer, RoboCup Server, multi-robot systems, ATAN Interface, JAVA.

INTRODUCCIÓN

ATAN, desarrollado por Wagner (2002), es una interface para el servidor de la liga de simulación 2D de la RoboCup, permitiendo concentrarse en el control de los clientes sin preocuparse en detalles de fondo, el lenguaje de programación a considerarse en ATAN corresponde a Java. Algunas de sus características son:

- Conexiones al servidor y al Soccer-Server por vía UDP.
- Análisis de las cadenas de comando que vienen desde el Soccer-Server.
- Generación de cadenas de comando que puede ser entendido por el

Soccer-Server.

- Conversión del lado de juego de un equipo, no es necesario preocuparse de qué lado del campo un jugador juega.

Los jugadores del Soccer-Server se comunican vía conexión UDP con programas externos que controlan al jugador indicando que acciones debe ejecutar el jugador, estos programas externos se comunican con el servidor usando una interfaz de cadenas que son descrita en el manual del servidor, Chen et al. (2003), ver Figura 5.10.

ESTRUCTURA DE LOS COMPORTAMIENTOS MEDIANTE LA INTERFAZ ATAN

Básicamente puede verse tres componentes mediante los cuales se construye y se ejecuta un equipo robótico de futbol: construcción de los comportamientos de los jugadores, construcción del equipo y conexión del equipo al servidor.

Construcción de comportamientos

Se tienen dos interfaces ControllerPlayer y ActionPlayer.

En la interface ControllerPlayer, se observan los métodos que definen los estados inicial y final luego de hacer lectura de los sensores, ver Tabla 1.

Tabla 1: Métodos ejecutados antes y después de la lectura de los sensores

Método	Descripción
preInfo()	El método es llamado antes que el controlador reciba la información de los sensores.
postInfo()	El método es llamado después de procesar toda la información de los sensores.

Otros métodos definidos son los InfoHear permiten obtener información sobre los mensajes que envían otros actores del campo como el árbitro u otro jugador, ver Tabla 2.

Tabla 2: Métodos InfoHear

Método	Descripción
infoHearReferee()	El controlador es informado cuando el árbitro envía una señal.
infoHearPlayMode()	El controlador es informado sobre el modo de juego.
infoHearPlayer()	El controlador es informado cuando otro jugador envía un mensaje.
infoHearError()	El jugador es informado cuando escucha un mensaje de error.
infoHearOk()	El jugador es informado cuando escucha un mensaje de OK.
infoHearWarning()	El jugador es informado cuando escucha una advertencia.

Mediante los métodos InfoSee, se proporciona información del sistema visual del jugador, ver Tabla 3.

Tabla 3: Métodos InfoSee

Método	Descripción
infoSeeLine()	Una de las líneas de paso esta a la vista.
infoSeeBall()	El balón esta a la vista.
infoSeeFlagRight()	Una de las banderas a lo largo de la banda derecha está a la vista.
infoSeeFlagLeft()	Una de las banderas a lo largo de la banda izquierda está a la vista.
infoSeeFlagOwn()	Una de las banderas detrás del área de meta del propio equipo está a la vista.
infoSeeFlagOther()	Una de las bandera detrás del área de meta del otro equipo está a la vista.
infoSeeFlagCenter()	Una de las líneas centrales esta a la vista.
infoSeeFlagCornerOwn()	Una de las banderas de esquina del propio equipo está a la vista.
infoSeeFlagCornerOther()	Una de las banderas de esquina del otro equipo está a la vista.
infoSeeFlagPenaltyOwn()	Una de las banderas de penal del propio equipo está a la vista.
infoSeeFlagPenaltyOther()	Una de las banderas de penal del otro equipo está a la vista.
infoSeeFlagGoalOwn()	Una de las banderas de meta del propio equipo está a la vista.
infoSeeFlagGoalOther()	Una de las banderas de meta del otro equipo está a la vista.
infoSeePlayerOwn()	Un jugador del propio equipo está a la vista.
infoSeePlayerOther()	Un jugador del otro equipo está a la vista.

Un jugador también tiene características como su energía, la velocidad, dirección y demás que son proporcionados por el método infoSenseBody.

La interface ActionPlayer permite al jugador contar con métodos mediante los cuales se ejecuta acciones de manera similar a los actuadores o efectores de un robot.

En la tabla 4, se muestran las acciones ejecutadas por un jugador.

Tabla 4: Métodos para ejecutar las acciones del jugador durante el juego

Método	Descripción
move()	Mueve al jugador en una posición especifica del terreno de juego
catchBall()	Le permite coger el balón al arquero
dash()	Acelera el movimiento del jugador en dirección a su cuerpo
kick()	Acelera el movimiento del balón
turn()	Gira el cuerpo del jugador en grados respecto a la dirección actual
turnNeck()	Gira el cuello del jugador en grados
say()	Envía un mensaje por el campo de juego

Construcción del equipo

La Tabla 5 muestra los métodos definidos en la clase AbstractTeam de la interfaz Atan.

Tabla 5: Métodos de la clase AbstractTeam

Método	Descripción
connect()	conecta un jugador al servidor
connectAll()	conecta todos los jugadores al servidor
createNewPlayers()	crea un nuevo jugador
getNewController()	retorna un nuevo controlador
getPlayer()	retorna un jugador
getTeamName()	retorna el nombre del equipo
size()	obtiene el tamaño del equipo

Conexión del equipo al servidor

Mediante los métodos connectall() o connect() se hace las conexiones al servidor del simulador.

CARACTERÍSTICAS DEL CAMPO DE JUEGO

El terreno es descrito en el plano cartesiano con punto de referencia en (0,0) como punto central, ver Figura 1.

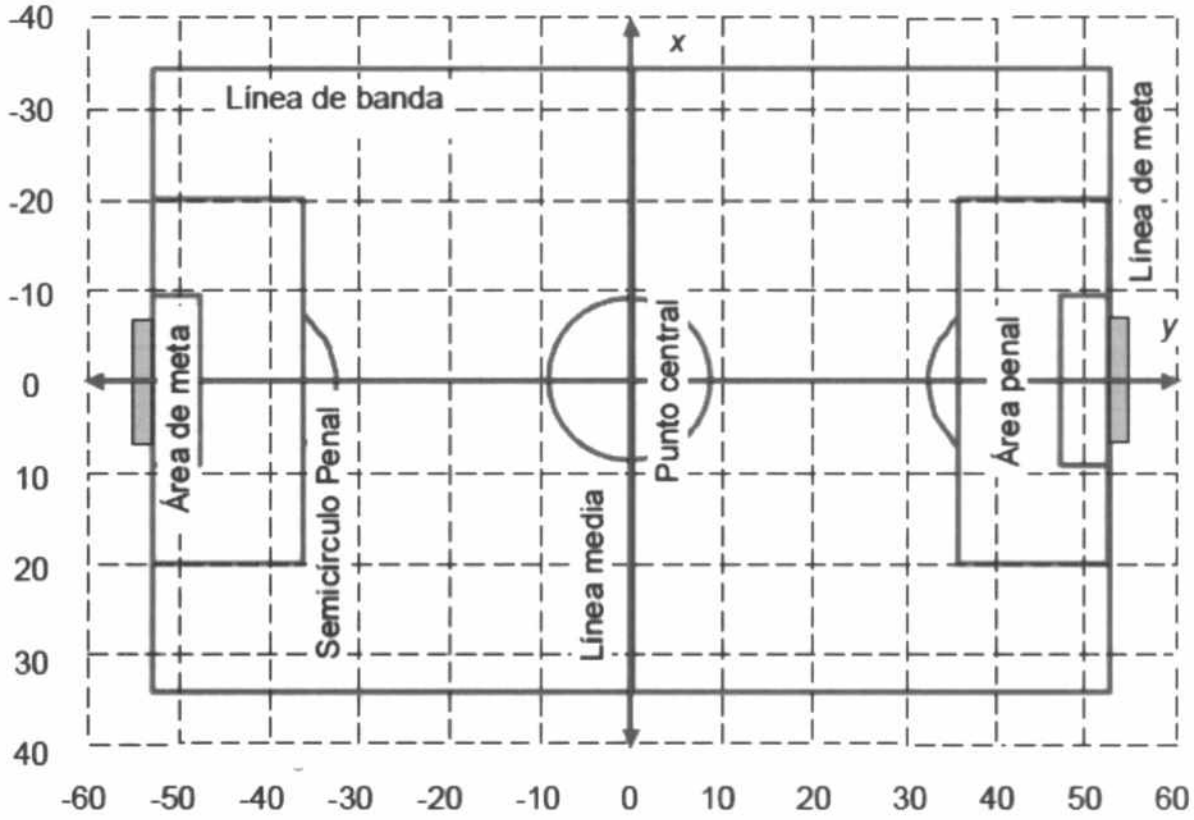

Figura 1: Terreno de juego definido en el simulador 2D de la RoboCup

Se asume que los valores en la escala están expresados en metros, así un jugador mediante el método move(x,y) puede moverse a cualquier posición del terreno de juego, donde $x \in [-52.5, 52.5]$, $y \in [-34, 34]$.

De esta manera en la figura 1, se muestra una aproximación de las dimensiones de las áreas del terreno de juego. En la Tabla 6 se muestra las dimensiones de los elementos del terreno de juego.

Tabla 6: Dimensiones de los elementos del terreno de juego

Objeto	Longitud	Anchura
Terreno de juego	105	68
Área penal	16,5	40,32
Área de meta	5,5	18,32
Portería	-	14,02

MODOS DE JUEGO RECONOCIDOS POR ATAN

Sobre la base del servidor de la RoboCup donde el servidor envía el inicio de un modo de juego mediante el árbitro automático, en Atan puede verse 26 modos de juego, que son controlados de manera automática por el árbitro. Así el árbitro envía mensaje a los jugadores indicándoles el modo de juego actual

PROGRAMAS CON SENSORES Y EFECTORES

En esta sección se analizaran ejemplos de programas bajo la interface Atan que manejaran la información recibida por los sensores y las acciones a ejecutarse mediante los actuadores o efectores.

Información del balón

En el Programa 1, describe la clase "equiporun" que hace la conexión del equipo Bolivar al servidor

```
import atan.model.AbstractTeam;

import org.apache.log4j.BasicConfigurator;

public class equiporun {

   public static void main(String[] args)

    { BasicConfigurator.configure();

      AbstractTeam team = new equipo("Bolivar",
6000, "localhost");

       team.connect(1);

    }

}
```

Programa 1: Clase equiporun

La clase equipo construye el equipo de juego denotado como "equipo", en función al jugador que sea conectado se efectúa la asignación del comportamiento asignado al jugador, así mediante el método getNewControler() que tiene como parámetro de entrada el número del jugador se retornó el comportamientos del jugador. Para el presente caso de un solo jugador conectado al servidor no considera el parámetro de número de jugador, ver Programa 2.

```
import atan.model.AbstractTeam;

import atan.model.ControllerPlayer;

public class equipo extends AbstractTeam {

  public equipo(String name, int port, String host-
name) {

      super(name, port, hostname);

   }

  public ControllerPlayer getNewController(int
number) {

         return new comportamiento();

   }

}
```

Programa 2: Clase equipo

En el Programa 3 se establece la posición inicial adoptada por el jugador antes de iniciar el juego mediante el comando "move", las coordenadas corresponden a la fila y columna descrita en la Figura 1.

```
public void infoHearPlayMode(PlayMode play-
Mode) {

 if (playMode==PlayMode.BEFORE_KICK_OFF)

     {getPlayer().move(-20, 10);

     }

  }
```

Programa 3: método infoHearPlayMode

El metodo "infoSeeBall" descrito en el Programa 4, permite obtener información del balón como dirección, posición y otros, este método se ejecuta siempre que el balón está en la vista del jugador, considerando que el balón ha sido registrado por el sensor visual del jugador, entonces se recupera la distancia y la dirección del balón en las variables: "distanciabal" y "direcciónbal", también se cambia el estado de la variable booleana "puedeVerBalon" a "true", esta variable ha sido inicializada con el valor de verdad "false", este cambio de "verdad" a "falso", permite indicar que el balón ha sido visto por el jugador. Estas tres variables son declaradas de manera global al inicio de la clase "comportamiento".

```
Public void infoSeeBall(doublé distance,
doublé direction, doublé distChange, doublé
dirChange, double bodyFacingDirection, dou-
ble headFacingDirection) {

        distanciabal=distance;

        direccionbal=direction;

        puedeVerBalon=true;

  }
```

Programa 4: Método infoSeeBall

El método postInfo() descrito en la programa 5, describe las acciones del jugador luego de recibir toda la información correspondiente a su entono de juego. Inicialmente se define la variable "puedeVerBalon" con una variable booleana inicializado con valor de verdad igual a "false", si la variable booleana cambia de estado por el Programa 4 entonces ingresa a la estructura condicional "if" donde gira con dirección al balón, avanza hacia esa dirección con velocidad de 15, en otro caso si el balón no ha sido visto el jugador gira 45 grados con dirección a las agujas del reloj.

```
public void postInfo() {
        if (puedeVerBalon)
            { getPlayer().turn(direccionbal);
              getPlayer().dash(15);
            }
            else
            {getPlayer().turn(45);
            }
}
```

Programa 5: Método postInfo

En la Figura 2 se muestra algunas imágenes del comportamiento del jugador en la tarea de buscar e balón y dirigirse a ella.

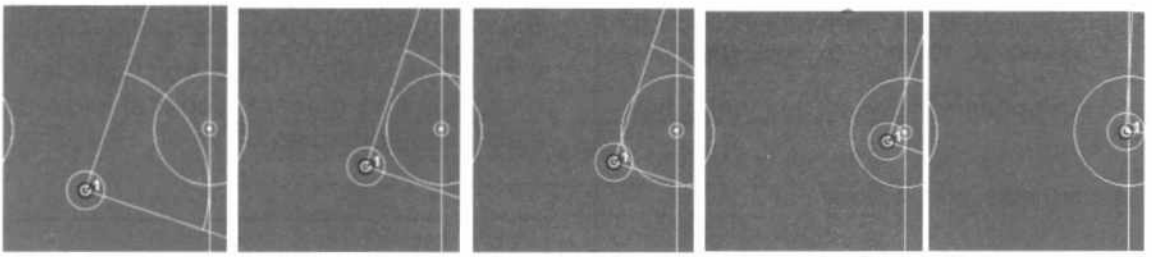

Figura 2: El jugador realiza una rotación en dirección al balón y se dirige a ella hasta hacer contacto con el balón.

CONCLUSIONES

El simulador de la RoboCup es una plataforma que cubre hasta los mínimos detalles en la simulación del futbol Robótico, varios concepto del campo de Ciencias de la Computación están inmersas en él como por ejemplo la programación paralela. Sistemas multi-agente, sistemas multi-robot, etc. Así esta plataforma también proporciona soporte para la experimentación de trabajos en esas áreas.

El servidor por defecto inicia con un árbitro automático que dirige el juego, enviando mensajes a través del servidor a todos los clientes sobre las faltas y modos de juego. Al respecto existen dos modos de juegos adicionales en los que puede arrancar el servidor: con el entrenador en line y el entrenador fuera de línea. El entrenador en línea puede interactuar con los jugadores durante el juego enviándoles mensajes, en cambio en entrenador fuera de línea no necesariamente puede participar durante un juego, este último modo de juego es muy útil para realizar pruebas de los programas asociados a los jugadores analizando los comportamientos en distintos modos de juego como ser los tiros libres, saque de meta y otros, particularmente en este trabajo se optó por este modo de juego: "entrenador fuera de línea", para experimentar y analizar las capacidades de los sensores y efectores del jugador. Así en realidad puede conectarse al servidor 11 jugadores más un entrenador.

BIBLIOGRAFIA

Atan (2012). *Atan*, recuperado de http://robocup-atan.github.io/atan/

Chen M., Dorer K., Foroughi E., Heintz F., Huang Z., Kapetanakis S.,...Yin X. (2003). *RoboCup Soccer Server*. For Soccer Server Version 7.07 and later. Febreary 11. Recuperado de http://sourceforge.net/projects/sserver/files/

Kitano H., Asada M., Kuniyoshi Y., Noda I., Osawa E., & Matsubara H. (1997). *RoboCup: A Challenge Problem for IA*. AI Magazine, 18(1).

Noda I., Matsubara H., Hiraqui K., & Frank I. (1997). *Soccer Server: a tool for research on multi-agent systems*. Applied Artificial Intelligence, 12(2), 233-250.

Wagner W. (2002). *Atan*, VSOC project, Vienna – Austria. Recuperado de http://atan1.sourceforge.net/