Cyberlords+Falconbots RoboCup 2012 Humanoid KidSize Team Description Paper

Luis F Lupián†, Alberto Romay†, Andrés Espínola†, Diego Márquez†, and Diego M Reyes‡

† Mobile Robotics and Automated Systems Lab, Universidad La Salle, México
lupianl@lci.ulsa.mx http://zeus.lci.ulsa.mx/lrm/
‡ Instituto Tecnológico Superior de San Martín Texmelucan

Abstract. We describe the RoboCup KidSize humanoid robots to be used by team *Cyberlords + Falconbots* in the RoboCup 2012 competition to be held in Mexico City, Mexico. For this edition of the competition we will present *five* different robot architectures. One of them is based on the Robonova-1 robot, two on the Kondo KHR-3HV, one on the Bioloid and the fifth one is a Darwin-OP. The main challenge for the team has been to have a common software architecture that will allow the robots to interact seamlessly. This challenge has been solved using a software library called *libCyberlords*, which is the main focus of this paper.

1 Introduction

Team *Cyberlords + Falconbots* is a joint team established in January 2012 for the purpose of competing at the RoboCup 2012 World Championship and the RoboCup Mexican Open 2012. The team is integrated by *Cyberlords La Salle*, which started participating in the RoboCup Humanoid KidSize League in 2008, and *Falconbots Tec San Martín*, which had its first participation in the same league in 2011.

Team *Cyberlords La Salle* , which is part of the *Mobile Robotics and Automated Systems Laboratory* at *Universidad La Salle* México, started its humanoid robot project in July 2008. The team debuted becoming champion at the *First RoboCup Mexican Open* in September 2008. Since then, the team has taken part in three *RoboCup World Championships*, two *RoboCup Latin American Opens* and two additional *RoboCup Mexican Opens*. The team has shown consistent improvement since its first *RoboCup World Championship*. In RoboCup 2009, team *Cyberlords La Salle* scored 2 goals and won one of its three matches. In RoboCup 2010, the team advanced to the second round and scored a total of six goals. In these two world championships the team participated in collaboration with the *Robotics and Artificial Vision Laboratory* of *Cinvestav* [1], [2]. In RoboCup 2011, the team's participated on its own, its top scorer Max scored seven goals and the team was just one goal away from reaching the quarter finals. At the *RoboCup Latin American Open* team *Cyberlords La Salle* has become champion for two years in a row, 2010 and 2011. In the most recent *RoboCup Latin American Open*, which took place in Bogotá, Colombia, our top scorer Max broke its own record by scoring nine goals in four matches (Fig. 1).

Fig. 1. Max at the *RoboCup Latin American Open*, Bogotá, Colombia, October 2011

Team *Falconbots Tec San Martín* is a young team that started working with humanoid robots in 2010. Its first participation at an official *RoboCup* event was the *RoboCup Mexican Open 2011*. In this event it obtained the third place out of seven participants. After that, team *Falconbots Tec San Martín* participated at *RoboCup 2011* in Istanbul, Turkey, where it tied one of its three matches.

2 Heterogeneous Hardware Architectures

The humanoid league distinguishes itself by the fact that every team strives on producing a hardware architecture that will give them an edge over opponent teams. On the one hand, although some teams may use some commercial robot as a starting point, such as Robonova-1, Kondo KHR-3HV, Bioloid, Darwin-OP, and so on, it is almost always necessary to perform significant hardware upgrades to allow these commercial humanoid robots to become autonomous soccer players, with the only notable exception of the Darwin-OP. On the other hand, there are teams that design their hardware architecture from scratch. The result is that almost every team in the humanoid league has a different hardware architecture, which makes it difficult for research groups to collaborate at the software level and even for robots from different groups to play together as a single team, which was made evident by the experimental 5 vs. 5 demo game held at the end of *RoboCup 2011* in Istanbul, Turkey.

This is why team *Cyberlords La Salle* believes there is a need for a software architecture that will allow humanoid robots with completely different hardware architectures to interact seamlessly in a RoboCup Soccer game. Our response to this need is a software library called *libCyberlords*. This is an object-oriented library that has been developed with interoperability of humanoid robots of heterogeneous hardware architectures in mind. This library has already been tested successfully at *RoboCup 2011* and *RoboCup Latin American Open 2011* in our top scorer robot Max. In *RoboCup 2012* we intend to show how flexible *libCyberlords* is by playing with humanoid robots of *five* different hardware architectures.

The details of the five hardware architectures for team *Cyberlords + Falconbots* can be found at the set of spec sheets for the team. To give a general sense of the differences between these architectures one of them is based on the Robonova-1 robot, two on the Kondo KHR-3HV, one on the Robotis Bioloid and the fifth one is a Robotis Darwin-OP, Fig. 2.

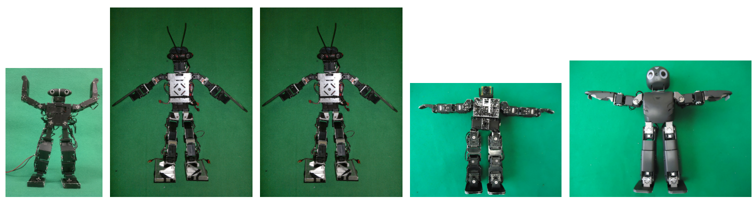

Fig. 2. Five hardware architectures for team *Cyberlords + Falconbots* in 2012

The following sections describe the main modules that are part of the *libCyberlords* library and allow it to be easily portable among hardware architectures.

3 Kernel and State Machine Metalanguage

At the heart of *libCyberlords* is the kernel, which provides three basic classes: Task, State and StateMachine, see Fig. 3.

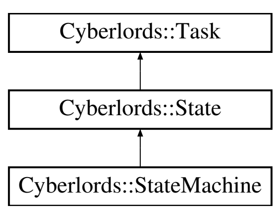

Fig. 3. Task-State-StateMachine hierarchy in *libCyberlords*

On the lower end is class Task, which provides the basic mechanism for low-level cooperative task concurrency from within the application. On the higher end is class StateMachine, which as its name implies implements a mechanism for construction of state machines. The class StateMachine is a descendant of the class State, this means that each state within a state machine can be in turn a state machine itself, and this allows for the construction of hierarchical state machines. The class State is a descendant of Task, so every state or state machine can execute concurrently with other tasks within the application. This has shown to be a very efficient and effective way of implementing concurrency and will be the subject of upcoming publications from our research lab.

One powerful feature of *libCyberlords* is a metalanguage that is used to construct state machines using simple transition tables. The class Transition is essentially a triplet composed of an origin state, a destination state and a condition. Condition is an abstract base class for any boolean condition. Listing 1 shows a sample code written using the *State Machine Metalanguage*, which represents a simple state machine for getting up after a fall. This state machine has only two states: INITIAL and GETUP. As long as the condition ROBOTFALLEN is true the state machine will transition to GETUP otherwise it exits. More complex behaviors can be constructed by defining simple state machines and then composing them into a hierarchy. For example, Listing 2 shows a state machine with five states, out of which four are state machines themselves: WALKTOBALL, RUNTOBALL, AIMATGOAL, KICKATGOAL. Listing 2 also shows more complex high level conditions, which are also an important feature of *libCyberlords* and are the subject of the next section.

Listing 1. GetUp State Machine

```
//   Origin      Destination  Condition
// +---------+-----------+-------------------+
RT( INITIAL , GETUP,       ROBOT_FALLEN        );
RT( INITIAL , EXIT ,       DEFAULT_TRANSITION  );
// +---------+-----------+-------------------+
RT( GETUP,    GETUP,       ROBOT_FALLEN        );
RT( GETUP,    EXIT ,       DEFAULT_TRANSITION  );
// +---------+-----------+-------------------+
```

Listing 2. Walk and get up

```
//   Origin        Destination  Condition
// +-----------+-----------+-------------------------------------------------+
RT( INITIAL ,    WALKTOBALL, PERCEIVE_BALL && BALL_D(Near)                       );
RT( INITIAL ,    RUNTOBALL,  PERCEIVE_BALL && BALL_D(Far)                        );
RT( INITIAL ,    AIMATGOAL,  PERCEIVE_BALL && BALL_D(OnFeet) && !GOAL_O(Front));
RT( INITIAL ,    KICKATGOAL, PERCEIVE_BALL && BALL_D(OnFeet) && GOAL_O(Front)  );
RT( INITIAL ,    INITIAL ,   DEFAULT_TRANSITION                                  );
// +-----------+-----------+-------------------------------------------------+
RT( WALKTOBALL,RUNTOBALL,    PERCEIVE_BALL && BALL_D(Far)                        );
RT( WALKTOBALL,AIMATGOAL,    PERCEIVE_BALL && BALL_D(OnFeet) && !GOAL_O(Front));
RT( WALKTOBALL,KICKATGOAL,   PERCEIVE_BALL && BALL_D(OnFeet) && GOAL_O(Front)  );
RT( WALKTOBALL,INITIAL ,     DEFAULT_TRANSITION                                  );
// +-----------+-----------+-------------------------------------------------+
RT( RUNTOBALL,  WALKTOBALL, PERCEIVE_BALL && BALL_D(Near)                        );
RT( RUNTOBALL,  AIMATGOAL,  PERCEIVE_BALL && BALL_D(OnFeet) && !GOAL_O(Front));
RT( RUNTOBALL,  KICKATGOAL, PERCEIVE_BALL && BALL_D(OnFeet) && GOAL_O(Front)  );
RT( RUNTOBALL,  INITIAL ,   DEFAULT_TRANSITION                                   );
// +-----------+-----------+-------------------------------------------------+
RT( AIMATGOAL,  KICKATGOAL, PERCEIVE_BALL && BALL_D(OnFeet) && GOAL_O(Front)  );
RT( AIMATGOAL,  INITIAL ,   DEFAULT_TRANSITION                                   );
// +-----------+-----------+-------------------------------------------------+
RT( KICKATGOAL,INITIAL ,    DEFAULT_TRANSITION                                   );
// + ----------+-----------+-------------------------------------------------+
```

4 Device Abstraction

Between two humanoid robot hardware architectures what is fundamentally different are the devices and their specific use for the task at hand. However, the high level behaviors are fundamentally the same. This means that if we create an abstraction layer for the numeric data produced by the sensors and for the motions of the actuators at the highest level behaviors can be easily ported from one architecture to another.

libCyberlords contains two essential components to achieve this device abstraction layer: SemanticPerception and MotionBehaviorEngine. The SemanticPerception module defines a number of abstract descendants of the Condition class that can directly be used for decision making while hiding the details of how the condition was evaluated. Examples of these semantic conditions are: RobotFallen, SeeBall, BallDistance. Each hardware architecture implements concrete descendants of these abstract conditions to be used in state machine construction. In the same way, there is a small set of abstract motion behaviors that can be defined for every humanoid robot, regardless of its physical construction. Example of abstract motions are: IniLftStepFwd, FaceUpStandUp, KickRgt, SitDown. For each humanoid robot architecture a concrete descendant of class MotionBehavior is implemented to specify the details of how to communicate with the set of actuators to achieve each specific motion.

5 Concrete Architectures

For each new humanoid robot hardware architecture a specific architecture module has to be developed. The architecture module essentially implements concrete descendants for the device abstraction layer. So for example, the condition RobotFallen could be implemented as a result of an interpretation of the data coming from the accelerometer, and would be highly dependent on how that accelerometer is physically installed on the robot's body. However, once implemented, this concrete descendant effectively hides the details of the hardware architecture and allows the same high level state machine to be seamlessly ported among humanoid robots.

6 Vision and Localization Subsystems

In this section, we describe team *Cyberlords La Salle* approach to vision and localization.

Localization of the robot within the field is achieved by extracting only fourteen landmark features that come from the corners of the goals and the segments of the landmark poles. To extract the features, the image is processed in three steps. The first one is the segmentation of colors blue and yellow. This is done by restricting the color 3D space to a minimum and maximum value along each of the three axes of the HSV color space [3], so the segmentation for each color requires only six calibration parameters. The output of the segmentation step is a pair of binary images, one for each color. The result of this process is exemplified in Fig. 4.

In the second step, the algorithm searches for contours in each binary image, enclosing the independent objects and producing at the output a list of objects

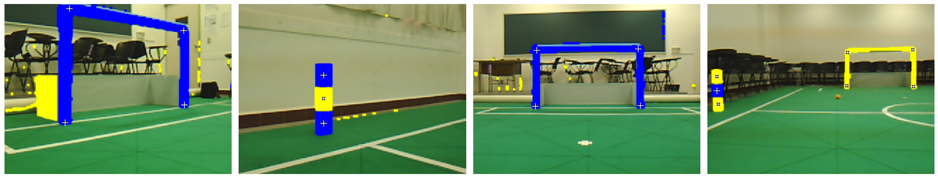

Fig. 4. Color segmentation and feature extraction

for each color, saving the information of the corners of the rectangle that contains all the pixels of that object.

For the third step, the resulting lists of objects from the previous step are analyzed in search for poles and goals. Landmark poles are searched for first. For the blue-yellow-blue pole, we select the first object in the yellow list and we search for the upper and lower blue segments in the blue list at the same distance up and down. If these match, the pole is accepted and the objects are removed from their respective lists. In case there is no match, we move on to the next object in the yellow list and repeat the same process until there are no more objects to analyze. An analogous process is applied for the yellow-blue-yellow pole. The cylindrical pole segments look almost rectangular from every point of view. The centroid of each of these perceived segments is used in our methodology to represent the image projection of the center of mass of that segment. This incurs in a small but tolerable measurement error for those cases when the pole segment projection is close to the top or bottom of the image.

After the pole search is complete, there is only the possibility of having the goals and other objects in the lists. For the goals, we made a small database of Hu moments [4] for the three different situations of the goals: only one goalpost, partial goal and complete goal, as shown in Fig. 5. We start to analyze the remaining objects in the lists, computing the Hu moments for each of them and comparing these moments to the ones in the database. Whenever a match occurs the corresponding object is analyzed in search for corners. The kind of matching Hu moments determines the kind and number of corners to look for. Figure 4 shows a sample of features extracted with the described methodology.

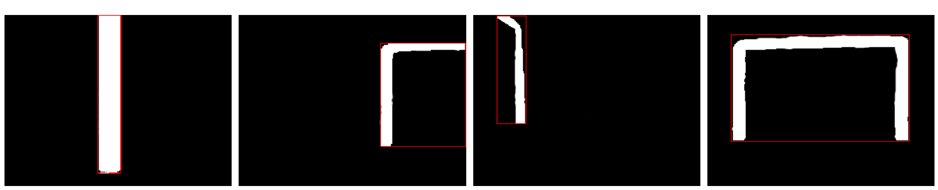

Fig. 5. Possible situations in goal perception

Once a list of observed features is available, localization is solved by formulating a regression model based on the pinhole camera projection model, as described in [5].

7 Conclusion and Future Work

We have described the hardware and software features of the humanoid robot architectures to be used by team *Cyberlords + Falconbots* in the RoboCup 2012 world championship. Our main contribution this year is a software library called *libCyberlords* which is the basis for interoperability among heterogeneous groups of humanoid robots.

We are currently working on developing the concrete architecture modules of *libCyberlords* for three of the humanoid robot architectures that team *Cyberlords + Falconbots* will bring to *RoboCup 2012*.

As we approach the vision of the *RoboCup* initiative for 2050, more complex and expensive robots will be needed. We will soon reach a point in time when no single research lab will be able to build, maintain and transport around the world eleven humanoid robots to play either against a similar team of humanoid robots or against a team of humans. The *RoboCup Humanoid League* will have to evolve into an event in which researchers will meet to integrate a mix and match of different kinds of humanoid robots from different groups around the world, and these humanoid robots will have to collaborate in a soccer game in a seamless way. The research groups participating at the *RoboCup Humanoid League* should start developing the necessary technical tools for this to happen, and at the same time, if we want the *RoboCup* vision to be accomplished, we should all start building the models for collaboration among research groups that will allow this evolution of the league to take place naturally.

Acknowledgements

This project is supported in part by the *CONACyT*, the *Mexican Robotics Federation* and the *Red Nacional de Robótica y Mecatrónica*.

We wish to extend our appreciation to Chris Vo from team *RoboPatriots* for the development of LIBKONDO4, which we've been using since March 2011 for the purpose of communication with *Kondo* servo motors through the RCB4 motor controller. His translations of the *Kondo* manuals from Japanese into English have also proved to be incredibly helpful for our team.

Team Members

Team *Cyberlords + Falconbots* for 2012 will be integrated by at least the following people:

- Team leader: Prof. Luis F. Lupián.
- Team members: Alberto Romay, Andrés Espínola, Diego Márquez, Iker Sanz, Omar Nelson, Francisco Lecumberri, Diego M. Reyes, Edgar Franco Pérez.

References

1. Lupián, L.F., Romay, A.I., Monroy, P., Espínola, A.F., Cisneros, R., Benítez, F.: Cyberlords RoboCup 2009 Humanoid KidSize team description paper. In: RoboCup World Championship, Graz, Austria, RoboCup Federation (July 2009)
2. Lupián, L.F., Romay, A., Espínola, A., Cisneros, R., Ibarra, J.M., Gutiérrez, D., Hunter, M., del Valle, C., de la Loza, K.: Cyberlords RoboCup 2010 Humanoid KidSize team description paper. In: RoboCup World Championship, Singapore, RoboCup Federation (June 2010)
3. Schwaez, M., Cowan, W., Beatty, J.: An experimental comparison of rgb, yiq, lab, hsv, and opponent color models. Graphics, ACM Transactions on **6**(2) (April 1987)
4. Hu, M.K.: Visual pattern recognition by moment invariants. Information Theory, IRE Transactions on **8**(2) (February 1962) 179 –187
5. Lupián, L.F., Romay, A., Espínola, A.: Vision Based Localization of Humanoid Robots by Inverse Pose-Estimation Using a Small Set of Fixed Landmark Features. In: Robotic Symposium, IEEE Latin American, Bogotá, Colombia (October 2011)
6. Lupián, L.F., Romay, A., Espínola, A., Ramírez, E.: Cyberlords RoboCup 2011 Humanoid KidSize team description paper. In: RoboCup World Championship, Istanbul, Turkey, RoboCup Federation (July 2011)
7. Reyes Hernández, D.M., Franco Pérez, E., Barbosa Segura, J., Sánchez Rodríguez, J.V., Suárez Romero, A., Méndez Lara, H.G.: Falconbots RoboCup 2011 Humanoid KidSize team description paper. In: RoboCup World Championship, Istanbul, Turkey, RoboCup Federation (July 2011)
8. Romay Tovar, A.I.: Visión computacional para problemas de estimación de pose. Master's thesis, Universidad La Salle, D.F., México (2011)
9. Espínola Auada, A.F.: Sistema de visión de un robot humanoide en un ambiente semi estructurado. Master's thesis, Universidad Nacional Autónoma de México, D.F., México (2011)

January 29th, 2012

Humanoid League Technical Committee,

I hereby express the full commitment of team *Cyberlords+Falconbots*, which I formally represent, to participate in the 2012 edition of the *RoboCup World Championship* to take place in Mexico City on June 18th – 24th.

I also express the full commitment of the team to provide at least one team member with sufficient knowledge of the rules to act as referee during the competition.

M. en C. LUIS FERNANDO LUPIÁN SÁNCHEZ
Profesor Investigador, Laboratorio de Robótica Móvil y Sistemas Automatizados
Facultad de Ingeniería, Universidad La Salle, México, D.F., MEXICO
+52(55)52789500 ext. 2275, lupianl@lci.ulsa.mx