An alternative approach to compete in RoboCup

Janjai Bhuripanyo, Rerngwut Choomuang
Department of Computer Engineering & Department of Mechatronics
Sripatum University (SPU)
Bangkok, Thailand

janjai.bh@spu.ac.th, rerngwut.ch@spu.ac.th

***Abstract*—This paper presents an alternate approach to prepare a RoboCup team by building a generic software simulator which is flexible enough to cover every robot configurations out there including their strong and weak points based on their configuration, their extraordinary features, information regarding to their hardware, their formation of attack and defense, etc. and many others. With the simulator the user can create a team of his/her choice, allow the simulator to take it's time try to find a proper team (configuration and combination) and a proper plan to play against other teams then review the all the outputs before preparing the user team accordingly. With this approach, the user team can face a few more teams through simulation for the whole year before the actual tournament comes around.**

***Keywords*—Software simulator**

I. INTRODUCTION

THE concept of software simulator has been around since 1970+, it has been utilized in all sectors from classroom, laboratories, research centers, automation industries all the ways up to the space industries. The drastic drop of computer pricing is another key factor which makes software simulator available at all levels. The obvious impact is a major reduction of the number of processes in developing new products or new invention. New design can be fully developed and tested on computer before they need to build the prototype. The old way which kept on building prototypes and tested until you can achieve your goal or ran out of your budget is gone forever.

RoboCup is an international event in which most of the top-rank universities/institutions around the world participate yearly. All the above competitors have put a lot of budget, times, resources and efforts, etc., to stay in contention and go for the top spot.

Local RoboCup event is in the same situation but at a smaller scale, local team within the same country try to compete for the top spot of their own to earn a right to compete internationally. Each year, local team with limited resources tried their best to look up to all the big boys (like Carnegie Mellon University, Cornell and lately with the addition of the mighty Plasma-Z from Chulalongkorn University of Thailand) and try their best to match with the above teams in term of the hardware and software. The biggest obstacles is in continuity of their resources especially manpower and development plan.

The main problems with most of the local teams can be summarized as follow:

- Limited resources in the areas of budget, time and manpower. This in turn lead them to find a quick solution to allow them compete with other teams. They try to match the hardware with the big boy teams but the hardware keeps on changing each year which force them to spend more time on the hardware itself.
- Any modification on hardware is likely to have impact on the software. Modifications on software are required to support new hardware. This cycle repeats itself each year and take away the rest of the efforts to improve the rest of the system.
- Practicing with other teams is likely to be zero because they spend most of their time to prepare their team rather than playing with other teams to find any weak spots on the team itself.
- There are too many competitors in the tournament and it is very difficult to formulate strategies for each team. The only thing they can do (for some teams) is to come up with a game plan and hope for the best that it will work for them against any teams in the tournament.

This paper proposed an alternative which can be carried on with limited resources as part of the development plan. The approach of software simulator will eliminate the trial and error processes in building robot with has occurred year in and year out for some teams. Good simulator will allow the user to decide and select a particular type of robot or a combination of various types of robots for his/her team to compete in the tournament. The output from the simulator may provide adequate data to allow the user to come up with appropriate algorithm for his/her team which based on the characteristic of the selected robots.

The simulator allows the user to simulate performance of his/her team to find any weak points so that the user and modified his/her algorithm accordingly. If the simulator has adequate information regarding to the opponent's team, the user can then use the simulator to simulate his/her team against the opponent's team (including their formation of attack and defense). This hopefully will increase his/her chance when the user actually plays against the opponent in the actual tournament.

II. Related Work

There are a lot of researches, released document [1] and papers [2], [3], [4], [5] on the RoboCup on both the small-sized and middle-sized league on the internet. The released document and papers provide both the inside information (on both hardware and software), problems and how to eliminate them, various algorithms, comparison between different algorithms including the strong points and weak points of each algorithms and their performances. The information is not only limited to global vision RoboCup teams but the local vision RoboCup teams as well (a few paper on small-sized but plenty of papers on the middle-sized RoboCup). Information from the above sources and direct contact to obtain inside information from some teams are also possible to make the input data to the simulator more realistic.

III. Proposed Software Simulator

The proposed software simulator has been developed according to rules and regulations of RoboCup. Only minor temporary changes are allowed to speed up the development processes but the above changes will be corrected accordingly in the near future. The following is the ground work for the proposed software simulator, they are:

1. Creation of various type of robots configuration and allow the user to utilize them during simulation run. Any modifications on the robot configuration must be done before the simulation run.

2. User can create an autonomous or non-autonomous team on the same playing field during simulation. This includes the position and heading of each robot on each team for the simulation run.

3. User can create global vision team against another global vision tem, global vision team against local vision team and local vision team against another local vision team.

4. The user can view/save/rerun simulation data such as: passing command protocol to each particular robot in the global vision environment, or passing command or message protocols between robots within the same local vision team, etc. Log file(s) can be created for the user.

5. Select certain type of algorithm for the team such as: Vector Force Field. At the moment, a generic algorithm similar to the Vector Force Field is the only algorithm supported by the software simulator but more algorithms can be added later on.

IV. Phase I (Hardware information)

Since the approach emphasis on software application, there is no hardware that needs to be designed, only information regarding to type of robot, characteristic, dimension of the robot, hardware on the robot, specific capabilities, etc., are needed. Example of the information are as follow:

1.Type of robot: 2-driving wheels with one supporting wheel, 3-driving wheeler or 4- driving wheeler including the angles between each wheel and related mathematical equations [1].

2.Type of wheel: omni-directional wheel (including number of the small rollers on each wheel), normal circular wheel including the diameter of the wheel, etc.

3.Dimension of the robot: height, width and length (in case of a rectangle shaped robot) or height and radius (in case of a circular shaped robot).

4.Type of vision system: Global vision, Local vision (in case of local vision, addition information are needed such as number of cameras and their locations on the robot, type of lens on each camera, area in which they are covered, height of camera on top of the robot and the circular top-view area that it covered, number of infrared sensors around the robot for obstacle avoidances, etc.)

5.Information regarding to the DC driving motor for each individual wheel: torque, rpm, encoder resolution, power consumption, etc.

6. Information regarding to the DC dribbling motor which controls the ball: torque, rpm, encoder resolution, power consumption, etc.

7. Information of the kicker: maximum/minimum impact force provided or how fast the ball can travel at maximum speed in case of shooting, how fast the ball can travel in some other condition such as passing, etc. This also includes related solenoid information.

8. Information of the drop kick mechanism over opponent robot: projectile information, range, distance needed to allow drop kick to work, etc.

9. Information regarding to the CPU and its processing speed, etc. This is another factor which will be supported by the simulator to allocate appropriate time slice for each individual robot during the simulation especially for the local vision team.

10. The image pattern on top of the robot which the opponent use or likely to use, etc.

11. Other related information, etc.

Most teams are willing to provide information regarding to their team during the tournament. Due to the length of the tournament, it is a little bit difficult to obtain all the necessary information but more information allows the simulator to closely imitate the opponent robot. The result of the simulator may lead to the weakness or limitation of the opponent in various situations on the playing field.

The above information along with other related database will be built into database for each particular robot. Variation of such information will be kept as another robot in the same database. It is up to the user to create any particular robot of his/her own choice and retrieves them for execution during the simulation run. This approach allows creation of any autonomous or non-autonomous team of robots on the same playing field.

V. PHASE II (SOFTWARE DESIGN)

The software simulator is written with C++Builder 6 from Borland Software Corporation. For simplicity, the simulator utilize ODBC available on C++Builder 6 to connect to Microsoft Access database using typical SQL commands.

The software simulator has been divided into 6 major portions, they are as follow:

1. GUI and supporting graphic functions. The graphic is in 2D, the playing field is proportional to the actual playing field including the size of the goal, pole, and related lines. For simplicity, 3 different colors (red, white and blue) will be assigned to each pair of poles, this allow the poles to act as landmark when local vision team is on the playing field.

2. Algorithms, this allow the user to select his/her algorithm from the existing poll of algorithms. At this time only one algorithm is being implemented, it is a generic algorithm similar to the Vector Force Field algorithm but the design provides option to allow the user to supply additional algorithms but it has to follow some guideline which allows the rest of the software to recognize the new algorithm.

3. Mathematical functions, this portion supports the basic mathematical models of various type of 3 types of robot, 2- driving wheel with one supporting wheel, 3-driving wheeler and 4-driving wheeler. The calculation for omni-directional wheel is not support at this time but they will be including in the calculation later on. There are other basic mathematical functions as library for the user to include into his/her interface to this software simulator.

4. Data communication function calls, these allow the user to send/receive typical structured formats of the communication protocol. This structured format contains basic data required for current simulation run, it can be added more data to the existing structure later on.

5. Main program handles all activities on the simulator, such as: data entry to create the required robot, create a team of his/her choice by selecting required robot from existing database, select options on algorithms, position and set heading of each robot on the playing field, assigned/unassigned specific duty to the robot (for example, goal keeper, defending robots, attacking robots, the assignment will put a boundary limit on the area that robot can move), local vision or global vision, etc.

6. Database's definition, shared variables and buffers. This portion allows the developer to receive all the necessary data to fit his/her needs.

Fig.1 shows a typical structure of the proposed software simulator. There are 3 main screens on the software simulator and many sub-screens beneath them. Fig.2 shows a typical main screen on the software simulator. Within this screen, the user can select the other 2 main screens: the Setup new robot's characteristic information screen and the Setup Team Assignment screen via the Setup Robot button and Build Team button respectively. Fig.3 and Fig.4 are typical Setup new robot characteristic information screen and typical Setup Team Assignment screen.

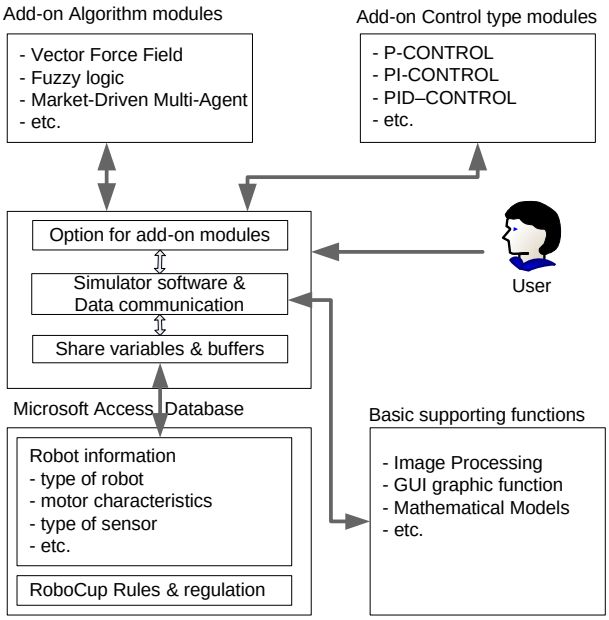

Fig.1 Typical structure of proposed software simulator

Fig.2 Typical Main Screen on the Software Simulator

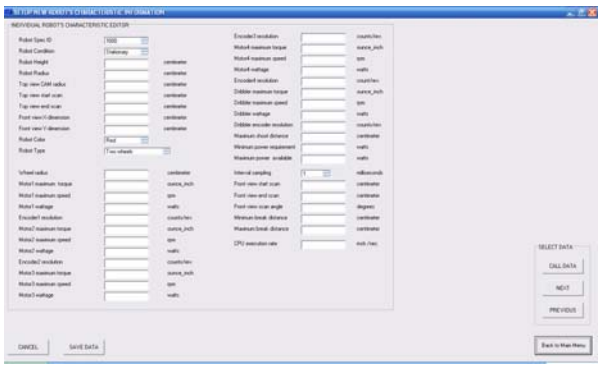

Fig.3 Typical Setup new robot characteristic information screen

Fig.4. Typical Setup Team Assignment screen

The Setup new robot characteristic information screen allows the user to build a new robot by selecting robot-ID and input the rest of the information regarding to the above robot into the database. Once the user finishes with the building process, the next step is to use the Setup Team Assignment screen to build a team of his/her own. By selecting the existing robots in the database, the user can have up to 5 robots on the team. The user can also set initial heading, assigns dedicated position (goal keeper, left back, right back, mid field and forward) and select control type for each individual robot. At the moment, the last two assignments are optional and will be implemented later on, these optional assignments allow the user to select different type of robots on the same team.

Once the building team processes finished (at least 2 teams must be built), the user can use the Main Screen on the simulation software to setup final initial conditions such as: selecting global/local vision capability, select team algorithm, which team starts the game, setup to monitor command/status, etc. The design team also includes future options such as: view simulation (allow the user to view any portion of simulation later on) and build algorithm (some type of learning AI program which can help user to generate appropriate algorithm for his/her robot team).

All the basic software has been developed and tested with some successes, the attacking team can generated some moves to avoid various obstacle on the playing field. Various spot testing indicated that the database has been utilized properly, all the shared variable and buffers contained all the required data. The message passing to/from the robots is working including the log file. The GUI functions provide the required information regarding to both the local and global vision.

Fig. 5 show various image processing data for overhead camera and front view camera of two local vision teams. The first picture show overhead image and front view image of the first robot (goal keeper)

on both team, the second and third pictures show overhead image and front view image of the second robot (back position, one robot in front of the goal keeper) and third robot (center half position, one robot in front of the back position) on each team respectively. The line within each robot indicates heading direction of such particular robot.

Fig.5. Various image processing data during moving test

VI. Test Methodology and Results

A. Functional Test

Each software functions/modules were developed from ground up, each of them was carefully tested by every member in the development team. This is necessary to make sure that they can provide good foundation for the application which will be on top of them in later phase of development. The tests show that there are some minor bugs, they have been corrected and all newly found bugs will be corrected as more tests progress along the way.

B. Performance Test

A series of simulation runs were performed for a period of time, data generated during the simulation runs were collected and reviewed accordingly. Both major and minor adjustment were make to the software to correct various problems which occurred in the application layers.

The simulation software provides a number of basic moves on a particular robot to move around the standstill obstacles (opponent team) toward the goal. Fig. 6. demonstrates a series of typical robot moves toward the goal. The overhead image and front view image of the robot changes based on the surrounding of the robot's position on the playing field.

Fig.6 Series of typical robot moves toward the goal

VII. Discussion

In general, the proposed software simulator finished with the first phase of software implementation with some additional fine tuning and documentation still need to done. The next phase of development will proceed as scheduled by the end of September 2007 and will concentrate in the following area:

1. Review the existing data dictionary to make sure that they are efficient and complete as much as possible. Current simulation runs indicate that there are some improvements that need to be done.

2. Update the missing mathematical models for the robot in the existing database and update related software accordingly.

3. Review the rule and regulation of the latest RoboCup in the database and make sure that they match with the current ones.

4. Replace the pole landmark with pole landmark used in the middle-sized league RoboCup and modify related software accordingly.

5. Enable real-time clock so that the user can control the amount of time being allocated to each team (or each individual robot in case of local vision). This is necessary to make the simulation very close to the actual event on the playing field in the tournament.

6. Allow the user to setup formation on each team during various points during the simulation.

7. Develop and tests additional algorithms, such as: Vector Force Field, Market-Driven multi-agent, etc. This allows user to pick and choose the appropriate algorithms for his/her team or for the each opponent team in the tournament.

8. Provide enable/disable feature for the image processing required in the global vision system to recognize each individual robot on the playing field. The feature wasn't implement in phase I.

9. Obtain more video information for the missing teams, extract the necessary video information (speed, shooting/passing range, attack/defense formation, etc.) and update the existing database for more simulation runs.

10. Additional researches in Multi-agents and apply them to local vision scheme. Only partial local vision was implemented in phase I, the robot in local vision scheme can make share information, make basic decision and move, coordinating between robots is very minimal.

VIII. Conclusion

There are still a lot of works to be developed and tested to make sure that this software simulator will work that way the development team set out to do. The team believes that this may be the fastest track and reliable ones for the team to keep up other teams and hopefully allow the team to play a few more rounds with many other teams and perhaps with the front-runner team(s) in RoboCup competition locally and else where in the future.

References

[1] Cornell University RoboCup: Big Red Bots. Final Design Document Team Brazil 1999. Prof. Raffaello D'Andrea & Dr. Jim-Woo Lee May, 1999.

[2] Distributed Control System on Small-sized RoboCup Jugkree Pala ka Wong Na Adyuthya, Chirit Chiritkhuan, Janjai Bhuripanyo, the 2nd IEEE International Conference on Cybernatics & Intelligent Systems & Robotics, Automation & Mechatronics. June 7-9, 2006. Bangkok, Thailand.

[3] Robocup Systems Engineering Project 2002 Liang-Yu (Tom) Chi, Wajih Effendi, Michael Jordan, Sin-Man Ko, Wei-Feng Li, Evan Malone, Shahab Najmi, Michael Schwaller. Project Advisor: Prof. Raffaello D' Andrea. May 2002.

[4] 2003 Cornell Robocup Document, Mechanical Group Final Document. Graham Anderson, Christine Chang, David Chung, Patrick Dingle, Leonard Evansic, Hank Law, Sean Richardson, John Roberts, Ken Sterk, Jeremy Yim. Project Advisor: Prof.Raffaelo D' Andrea, May 2003.

[5] Design of the ISePorto Robocup Middle-Size League Robotic Soccer Team: Control, Localisation and Coordination. MED2002 Conference A.Martins, J.Almeida, E.Silva, J.P.Baptista Institute of Superior de Engenharia do Porto, Portugal.

[6] Motion Control in Dynamic Multi-Robot Environments. Michael Bowling, Manuela Veloso. Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213-3890.

[7] Multi-Platform Soccer Robot Development System. Hui Wang, Han Wang, Chumiao Wang, William Y.C.Soh. Division of Control and Instrumentation, School of EEE, Nanyang Technological University, Nanyang Avenue, Singapore 639798. {p147513815, p149510969, eycsoh} @ntu.edu.sg

[8] Market-Driven Multi-Agent Collaboration in Robot Soccer Domain. Hatice Kose, Kemal Kaplan, Cetin Mericli, Utk Tatlidede & Levent Akin

[9] Robot Soccer Collision Modelling and Validation in Multi-agent Simulator. G.Klancar, M.Lepetic, R.Karba, and B.Zupancic. Mathematical and Computer Modelling of Dynamical Systems, 2003, vol.9, no.2, pp 137-150

[10] Multi-agent Control Structure for a Vision Based Robot Soccer System. Yangmin Li, Wai Ip Lei and Xiaoshan Li, Department of Electromechanical Engineering, Faculty of Science and Technology, University of Macau, Av.Padre Tomas Pereira J.J., Taipa, Macao SAR, P.R. China.

[11] Multi-Agent Fuzzy Control of the Robotic Soccer. Diniel Hladek, Technical University of Kosice, Slovakia, daniel.hladek@tuke.sk. 5th Slovakian-Hungarian Joint Symposium on Applied Machine Intelligence and Informatics, January 25-26,2007, Poprad, Slovakia

[12] Construction and Soccer Dynamics Analysis for an Integrated Multi-agent Soccer Robot System. Han-Pang HUANG, Chao-Chiun LIANG and Chun-Wei LIN. Robotics Laboratory, Department of Mechanical Engineering, National Tiwan University, Taipei, Taiwan, R.O.C. Proceeding in National Science Counc. ROC(A), Vol. 25, No.2, 2001, pp. 84-93.

[13] FU-Fighters Omi 2001 (Local Vision). Raul Rojas, Felix von Hundelshausen, Sven Behnke and Bernhard Frotschl, Free University of Berlin, Institute of Computer Science, Takustr.9, 14195 Berlin, Germany. http://www.fu-fighters.de {rojas|hundelsh|behnke|froetsch}@inf.fu-berlin.de