Botnia Dragon Knights - Team Description Paper for RoboCup 2008 Small Size League

Yang Liu[1], Rafal Chomentowski[1], and Tobias Glocker[2]

[1] Department of Information Technology
Vaasa University of Applied Sciences, Wolffintie 30, 65200 Vaasa, Finland
[2] Department of Computer Science
University of Vaasa, Wolffintie 34, 65200 Vaasa, Finland
yli@puv.fi

Abstract. This paper describes the Botnia Dragon Knights RoboCup Small Size League team, the current development status and some significant improvements to the previous years. It gives an overview of the new generation robot platform including the electronics, mechanics, strategy system, GUI, vision system, communication system, and simulator.

1 Introduction

The Botnia Dragon Knights team is a joint international RoboCup project and research team at Vaasa University of Applied Sciences, University of Vaasa in Finland and Hubei University of Technology in China, which consists of different student groups, i.e. electronics group, software group, mechanics group, telecommunication group and a staff group for steering the project as well as doing research. The primary goal of this project is to build a robot soccer team that can compete in the RoboCup tournaments, as well as many additional goals to promote the education and research. Botnia Dragon Knights joint team has been created in 2008 based on the formal Botnia team, which started in 2004 and has participated in the German Open 2005 (3rd place), Dutch Open 2006 (4th place), RoboCup 2006 (12th place). German Open 2007 (3rd place), RoboCup 2007 (9th place). The upcoming competitions we are heading for this year will be German Open 2008 and RoboCup 2008 in China.

The previous Botnia models showed many design and implementation flaws. Intensive investigations of the old design have been carried out. Improvements and redesigns are proposed and implementations are on the way. The major improvements over the old generation robot include precisely modeled mechanics, more advanced electronics, optimized power consumption, more stable kicking system, more reliable wireless communication, fully threaded strategy system with many new features and a newly built simulator. In the mean time the new generation robot has been completely redesigned with new mechanics and new electronics.

2 Overview of Botnia Robots

Botnia team has built three generations of robots and currently the Botnia Dragon Knights joint team is building the forth generation. The first generation Botnia SR1 in Figure 1 was built in early 2005 to participate German Open 2005. It was based on 3-omni-directional-wheel design using Faulhaber DC motors and 9.7:1 gearhead, with spinner bar kicker. The electronics involved Atmel ATmega microcontroller, discrete motor controllers, and Radiometrix RF module. It showed many design flaws during the competition and an enhanced version Botnia SR1E was built in early 2006 to participate Dutch Open 2006. SR1E used same platform as SR1 but eliminated major flaws, with major improvements of motor control and vision system. However due to a very weak kicker, SR1E was still far behind competitive during the games. After Dutch Open 2006, a hard decision was made to build completely new second generation Botnia SR2 to participate the forthcoming RoboCup 2006, with major goals to use solenoid kickers and to improve the drive system.

Fig. 1. Botnia SR1/SR1E

Fig. 2. Botnia SR2C/SR2D

Botnia SR2 was completely handmade due to the extremely limited time. From design to implementation it took just one and half month, with all days and nights' work. SR2 was still based on 3-omni-directional-wheel design using Faulhaber DC motors, with higher resolution encoders (128/rev compared to 16/rev used in SR1 series), enhanced 24-subroller wheel, and enhanced 9.7:1 ball bearing gearhead. The main circuit was completely redesigned with optimized

Fig. 3. Botnia SR3 prototype

high efficiency DC converter and noise reduction capability, with SMD 4-layer PCB. The power supply voltage was increased from 8-cell to 10-cell 2600 mAh Ni-MH battery pack. The wireless communication system was redesigned to be able to switch modules between 433 MHz and 869 MHz. A standalone kicker circuit was built to supply 300V high voltage and charge a 470uF capacitor to power the solenoid kickers. An additional IR ball detector circuit was used to trigger the kickers. For an original factory made solenoid without changing the winding, the forward kicker achieved ball speed of 5 m/s. The kicker circuit was totally isolated with optical couplers and together with self discharging circuit made it very safe against electric shock. Due to the lack of mechanical experience and time, SR2 was not able to combine the forward kicker, chip kicker and dribbler bar all together. Eventually two variant modes of SR2 were built, namely SR2C and SR2D as shown in Figure 2. SR2C was the model with forward kicker and chip kicker. SR2D was the model with forward kicker and dribbler bar. We managed to use SR2C and SR2D to play as a mixed team during RoboCup 2006. SR2C was proved to be a quite successful model and scored all goals during the competition. The main weaknesses of SR2 series were the fragile and slippery wheels (handmade with plastic material) along with non-precision handmade mechanics, and its weak wireless communication performance under heavy interferences.

After RoboCup 2006, Botnia team planned to build its third generation Botnia SR3 for RoboCup 2007 as well as enhanced SR2E for backup and testing purpose. The major improvements of SR2E included upgrading the whole mechanics with CNC-made precision parts, redesigned wheels for better grabbing and enhanced wireless communication. In addition the kicker circuit was considered to be redesigned to reduce its size. The power source was also considered to be replaced with high performance Li-Po batteries. The third generation SR3 design was modified several times due to the inexperience of the team members and the outcome was not able to fulfill the design targets. 3D CAD was first time used to design mechanical parts for SR3 but the manufactured parts were not precise enough to meet the design requirements. SR3 was based on 4-omni-directional-wheel design using Maxon brushless motors with external 4:1 gear transmission. The major improvements of SR3 were using completely new architecture of electronic design and strategy software. SR3 used ARM7 processor running RTOS with discrete motor drivers. It was designed with capability of processing part of intelligence on robot itself, i.e. extracting wheel vectors

from coordinates, autonomous lining up robot toward ball, logging function of status of robot (battery level, power consumption, travel distance, communication performance), etc, however such functions were not implemented in time and therefore left to 4th generation design. The kicker circuit was redesigned to supply 200V high voltage and charge altogether 8800uF capacitors to power the solenoid kickers. It was capable to switch maximum 200A current with 1024 grades fine adjustable kicking strength. At the full kicking strength, the forward kicker could achieve ball speed of 18 m/s and the chip kicker could achieve chip distance of over 5 m. The safety was also assured by using totally isolated circuit with optical couplers as used in SR2. The wireless communication in SR3 was significantly improved by using universal exchangeable modular design. SR3 could easily switch plug-in wireless modules between 1.9GHz DECT, 916MHz Linx, and 2.4GHz IEEE 802.15.4 radio. Multi antenna diversity was also taken into consideration to improve wireless communication performance. The prototype SR3 as shown in Figure 3, was tested in RoboCup 2007 with a major failure of its embedded software and incapable hardware. Nevertheless SR3 was later on completed after the RoboCup 2007 competition and used for educational demos.

Despite of the failure of SR3, Botnia has redesigned completely 4th generation SR4 in cooperation with the partner team in China. The SR4 is supposed to consummate all features designed in SR3 with additional improvements. SR4 still uses 4-wheel subsystem with a forward kicker, a chip kicker and a dribbler mounted on the platform. Modeling of the omni-directional robot is based on [3], [4] and [5]. The forward kicker mechanism of SR4 has been significantly improved with much more powerful full power kicking strength. The chip kicker mechanism of SR4 has been redesigned to reduce significantly the ball coverage percentage compared to previous versions. The mechanics are completely produced with CNC machines. The electronics improvements are significant over SR3. The main control system uses ARM7 and Altera EP1C6 Cyclone FPGA with extended data bus. All brushless motors are driven directly with FPGA plus MOSFETs solution. The wheel motors are equipped with 360 CPR optical encoders for high precision control. An IR array is implemented for self ball positioning. For wireless communications there are now both DECT and Linx modules onboard and no need to switch between each other. The power systems are optimized to achieve over 80% high effiecency DC step up/down convertion.

3 Mechanical Design

The new generation Botnia SR4 is designed based on the RoboCup F180 small-size league rules [1]. The height of the robot is 148 mm, and the maximum diameter of its projection to the ground is 178 mm, as shown in Figure 4. The maximum projected ball coverage distance is 10 mm as shown in Figure 5, which yields the maximum percentage of ball coverage of 17.65%. SR4 is currently still under development. Figure 6 shows a partly assembled SR4 prototype.

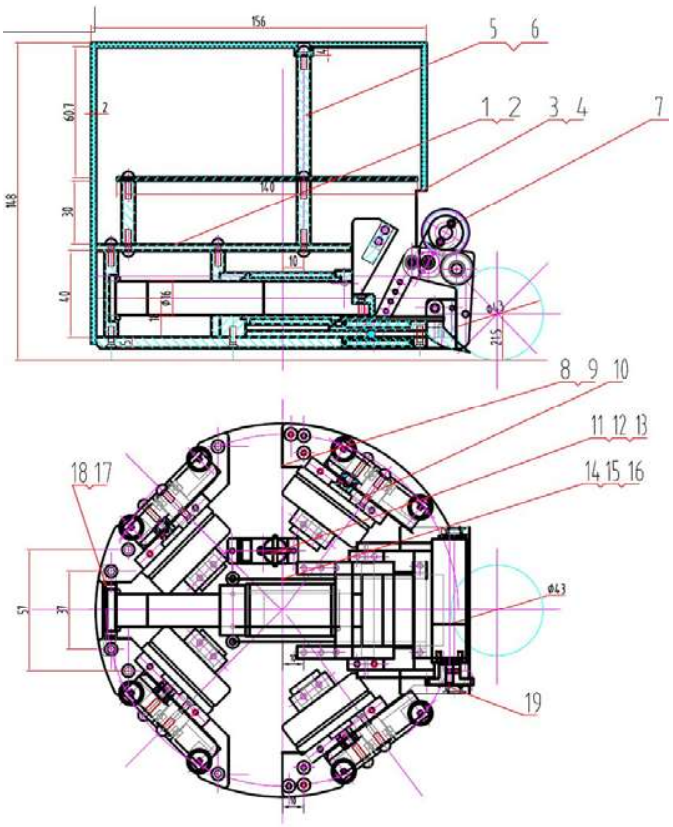

Fig. 4. Dimensions of Botnia SR4

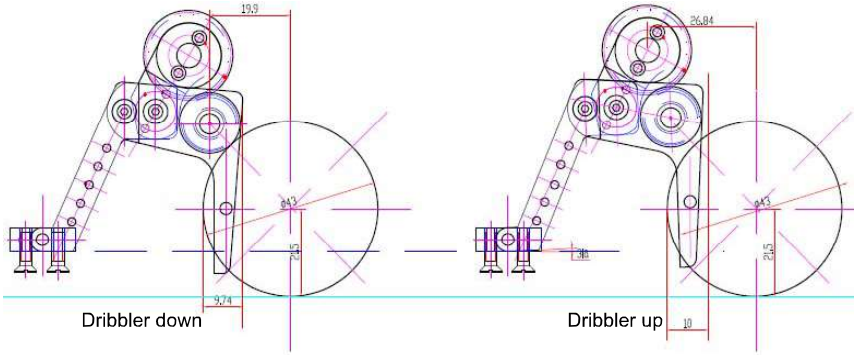

Fig. 5. Maximum ball coverage of Botnia SR4 at dribbler down/up position

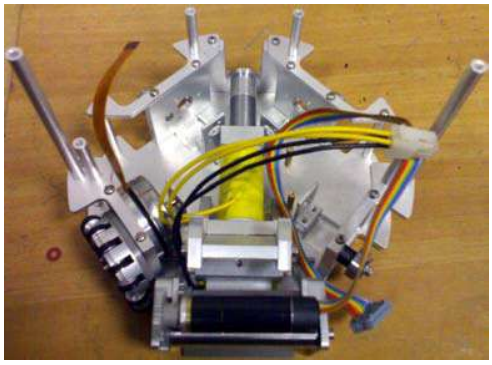

Fig. 6. A partly assembled SR4 prototype

4 Software Design

The overview of Botnia software structure is presented in Figure 7. It consists of strategy system, vision system, communication system, and a simulator which is still under development.

4.1 Strategy System

The old strategy system which was used to control SR1/SR1E/SR2C/SR2D has been completely overhauled. A gigantic loop was used to make the system work. This had lead to a very complex to manage system where adding functionality would mess up other parts of the system. Furthermore, the old design was not at all modular and would need a complete clean-up to be useful anyway. Also, the loop made the system quite slow and responses to events did not happen as fast as they should have.

The main goals for the new system include:

- Speed increase
- Modularity
- Expandability
- Testing of individual modules

Therefore a new design has been made as shown in Figure 8, which is both modular and expandable, yet easy to maintain with the ability to test individual parts of the system made easy. The solution was to make the system as threaded and concurrent as possible. To make actual use of these threads, a Sun T1000 CoolThreads server was acquired which has a SPARC CPU with 6 cores and 4 hardware threads per core. This makes it possible to run 24 threads in parallel.

In order to test modules separately, test software has been written on a per-module basis which allows various input and output states to be tested for

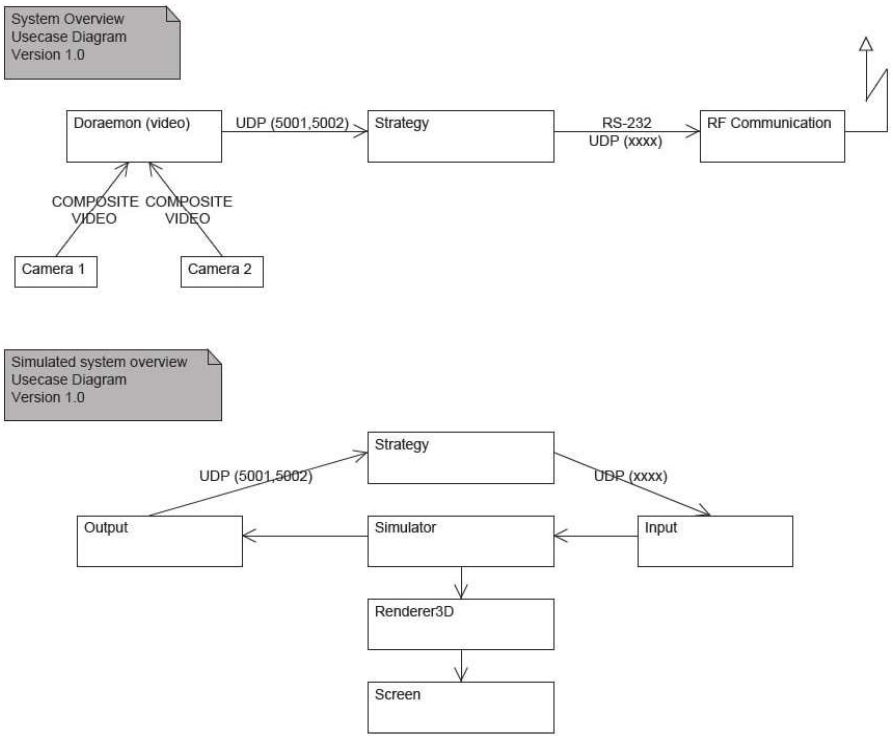

Fig. 7. Overview of software system and simulated system

both accuracy and performance. The Python language will be used to this end. Python scripts will be furthermore able to take control of the entire strategy server allowing for real-time interaction with the system during testing phases.

A further advantage of this system is that performance testing and optimization per module are possible, allowing people to test and optimize a module without having to know the entire system - or even have access to the entire system.

The strategy has been changed a few things from last year. The most important change is that it doesn't have ATTACKER or DEFENDER roles. Instead, it concentrated on assigning tasks most suitable for certain robot in a certain time. Robot rules are assigned in every frame. New tasks for robots are added, and at the moment there are 5 different tasks:

- Goalkeeper - it's just goalkeeper rule.

- TakeBall - it is a task assigned to the closest robot to the ball, tries to get the ball and if possible shoot on the goal or pass to other robot that have better position to shoot.

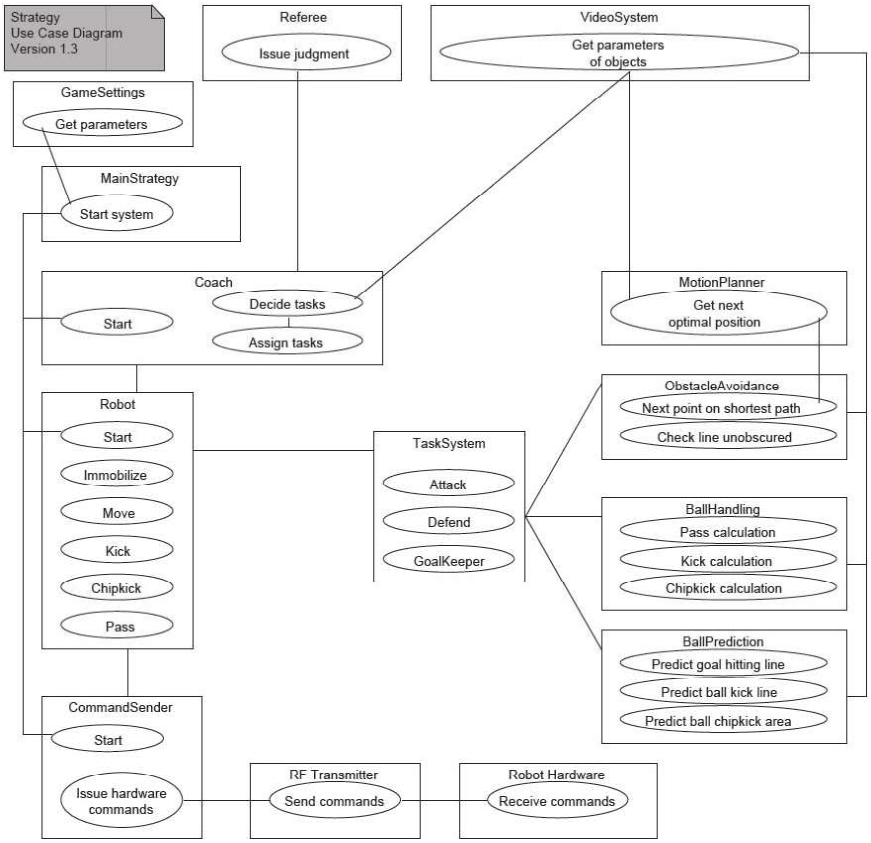

Fig. 8. State chart of the new strategy system

- ProtectGoal - the robot tries to cover the path to the goal as good as possible, tries to get to the opponent to a certain distance and blocks the path to the goal.

- Defend - covers free oponent robots, makes them unable to get a pass.

- FindClearSpot - goes to a position from where it can get a ball from our robot and have clear shoot path on the goal.

The Goalkeeper task is assigned beforehand, and it doesn't change during the game. TakeBall is assigned to the closest robot to the ball. Completely different things happen with other tasks. The strategy calculates cost of every task for every robot and then assigns tasks to robots, so the cost in total is lowest. This system makes sure that efficiency is very high, robots are doing less work, are faster, consume less power. The biggest advantage in the strategy system is that it isn't schematic. Positions aren't defined beforehand, and thus make it difficult to predict for opponent teams.

4.2 GUI

Botnia GUI is a software which was developed with Java (JDK1.6) to visualize the control of robots. This software has also the capability to record and watch recorded games, which is very useful for finding errors in algorithms and for the improvement of algorithms. It is a completely multi-threaded software which consists of 28 classes. Figure 9 shows the overview of the GUI.

The GUI is connected with two video servers, one for each camera. Each of them sends a plain text stream which includes the positions of the objects. Two UDP socket server threads that run within the GUI, parse the incoming plain text stream and write the positions of the detected objects (robots, ball) in a synchronized ring buffer. If both UDP threads have written their data into the buffer, a painting thread reads the data from the buffer and draws the objects on the football panel. In a MYSQL database positions of the objects will be stored for recording the games. The JDBC (Java Database Connectivity) driver is used for the database connectivity. Control of the robots is done over a TCP connection between the GUI and the strategy server. For the joystick control a library called LWJGL (Lightweight Java Game Library) is used. Java's common event listeners are used for the mouse, keyboard and button control.

Fig. 9. Overview of Botnia GUI

4.3 Vision System

The latest version of the Bt848178 driver for Linux Bttv2, V4L2 was used for the frame grabber. It provided a DMA streaming feature not available with older versions. Since the Linux X Windows system is based on a client-server method, Simple Direct Media Layer (SDL) [2] was needed to provide fast access to the graphics frame-buffer. Since an NTSC video standard output is used at 640x480 pixels with 30 frames per second the frame grabber can capture half of a frame at a time, even field / odd field i.e. 640x240 pixels at 60 frames per second. For such an output the processing cycle must be less than 16 ms in order to maintain the full frame rate.

The processing of such frames is done by the vision system as follows:

→ Read the configuration file and initialize the calibration data
→ Initialize frame grabber card
→ Initialize Video-server UDP socket
→ Repeat
→ Wait until one field of images data has been transferred to memory
→ Do image processing
→ Prepare the coordinate data and broadcast through the socket
→ Until the user stops the program

Apart from the main routine, a low priority thread was implemented to update the GUI. The status screen of the GUI display shows important information on the percentage of processing cycle used to find the object indicator for object found, the velocity of each object's movement and the frame rate.

Although the previous vision system was quite good over the past year, some improvements regarding speed have been made. Detection of all the objects has seen an increase of 30% in speed. For synchronisation purposes, the two video servers have been combined into one.

4.4 Communication System

In order to adapt to different communication hardware which has different data throughput, two different formats of control packet are introduced. With higher throughput communication modules, 10-byte control packet is used, including 2 bytes for robot control (3 bits for ID, 1 bit for forward kick, 1 bit for chip kick, 1 bit for dribbler, 10 bits for kicking strength), 2 bytes for X velocity, 2 bytes for Y velocity, 2 bytes for Angular velocity, 2 bytes for Error detection. On the other hand with lower throughput communication modules, 5-byte control packet is used, including 2 bytes for robot control (3 bits for ID, 1 bit for forward kick, 1 bit for chip kick, 1 bit for dribbler, 10 bits for kicking strength), 1 bytes for X velocity, 1 bytes for Y velocity, 1 bytes for Angular velocity.

Since the video frame capture rate using new cameras will be 50Hz, if the strategy is calculated based on each video frame then the control packet update won't be higher than 50Hz, but this is not good enough to control a fast process. Also there is always a delay between command sent to robots and their reaction. Nevertheless under most circumstances the variance of the delay is small, and it can be partly compensated by fine tuning the control parameters. The inter-frame control command is introduced, not only for sending commands but also receiving feedbacks, since there must be some kind of time duplex scheme used for transmitting and receiving, and they cannot happen simultaneously. Predication is highly beneficial to be used to decide inter-frame control commands, e.g. to predict where the ball will go and send robot control command already before waiting for the next video frame arrives. Therefore an overall update of control commands around 100Hz will be used to make good use of prediction. It's not really necessary to use full throughput of the communication module right now, but the capacity has been designed for the future.

Real-life performance of wireless communication has been intensively investigated in terms of round trip delay, bit error rate and packet error rate, etc, by interfering with simulated jamming signals, and results are shown in Figure 10.

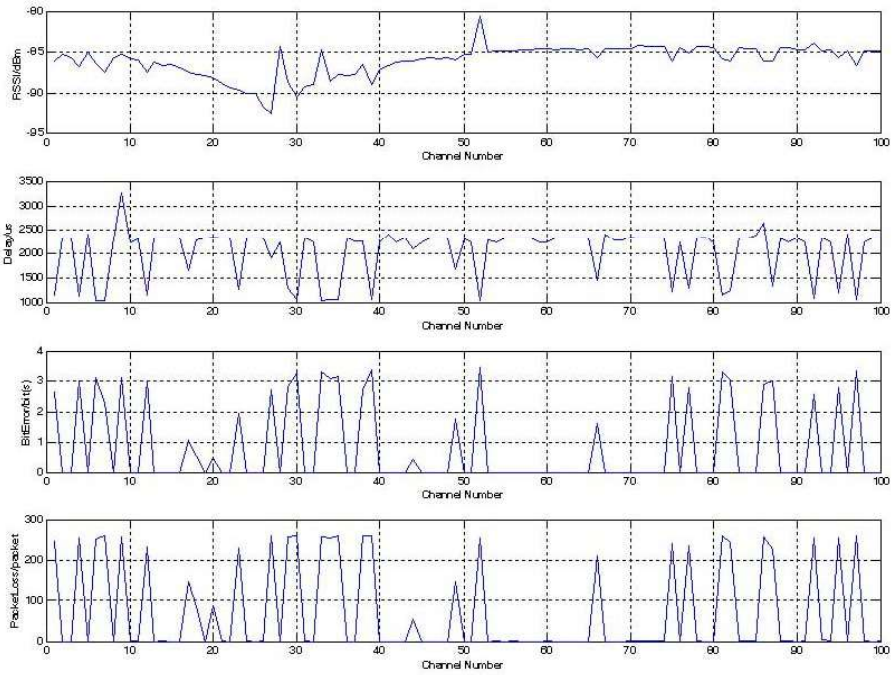

Fig. 10. Performance of Linx under interferences

The typical round trip time delay for Linx wireless transmission is around 209 us to 809 us. Under heavy interference the most common bit error rate is 3 bits per byte. Hence, a 3-bit error correction scheme should be sufficient to significantly improve the reliability of wireless performance. Besides error correction, frequency hopping is also employed to resist interferences. The RSSI levels of all channels are scanned and a black list is generated so that all the bad channels are excluded from the random frequency hopping sequence to ensure the hopping channel quality, and thus improve the communication performance even further.

4.5 Simulator

Extensive work is being put into the creation of an accurate simulator for testing purposes. The simulator will be able to make use of abstraction layers in order to use a variety of physics libraries and 3D engines. This allows for a flexibility regardless of the skills of individual developers. High end 3D engines allow for easy programming, while OpenGL allows for performance and fun.

5 Conclusion

This paper described the current development status of Botnia Dragon Knights team. A new generation of physical robots with completely new electronics, mechanics, communications as well as software have been designed and implemented, which have a number of significant improvements compared to last generation system. A software simulator is also currently under development.

References

1. RoboCup Organization Official Home page http://www.robocup.org [cited on 14.02.2007]
2. Simple Direct Media Layer (SDL) http://www.libsdl.org [cited on 14.02.2007]
3. Muir, P.F. and Neuman, C.P 1987: Kinematic modeling of wheeled mobile robots J. Robotic Systems, 4(2): 281-340
4. Sahn, S.K., Angeles, J. and Darcovich, J. 1995: The design of kinematically isotropic rolling robot with omni-directional wheels Mechanism and Machine theory, 30(8): 1127-1137
5. Y.P. Leow, K.H. Low and W.K. Loh 2002: Kinematic Modeling and Analysis of Mobile Robot with Omni Directional Wheels Proceedings of the Seventh International Conference On Automation, Robotics, Control And Vision, Singapore