GMD-Robots

Ansgar Bredenfeld, Thomas Christaller, Horst Guenther, Jörg Hermes,
Giovanni Indiveri, Herbert Jaeger, Hans-Ulrich Kobialka,
Paul-Gerhard Plöger, Peter Schoell, Andrea Siegberg

GMD - Institute for Autonomous intelligent Systems (AiS)
Schloss Birlinghoven, 53754 Sankt Augustin, Germany

e-mail: bredenfeld@gmd.de

1 Introduction

The overall research goal of GMD´s RoboCup team is to increase both, (1) the speed of mobile robots acting as team in a dynamic environment and (2) the speed of design for behavior-based robot control. Therefore, we started in 1998 to develop a proprietary fast robot platform and the integrated Dual Dynamics Design Environment.

In Melbourne, our middle-size league team achieved the second rank during round-robin. In the Quarter Finals our Robots were beaten by Sharif CE, the World Champion 1999 and current European Champion 2000.

2 Team

Ansgar Bredenfeld (DD-Designer, team leader)
Horst Guenther (technical support)
Jörg Hermes (ball guidance)
Giovanni Indiveri (control engineering of goalie)
Herbert Jaeger (Dual Dynamics)
Hans-Ulrich Kobialka (beTee)
Paul-Gerhard Plöger (Hardware)
Peter Schoell (DDSim)
Michael Severin (kicking device)
Andrea Siegberg (technical support)

Web Page: http://ais.gmd.de/BE

3 Hardware Platform

Our robot hardware is a custom-built platform. We use two 20 Watt, high-quality Maxon motors that are mounted on a very solid, mill-cut aluminium frame. A piezo-gyroscope senses the angular velocity of the robot. Obstacle avoidance is supported by four infrared-based range detectors and standard bumper ring sensors. Our robots kick the ball with a pneumatic device. The camera of the Newton Lab's Cognachrome vision system is mounted on a 360 degree panning unit.

The computer system of the robot consists of a Pentium PC notebook connected to two C167 micro controller subsystems for sensor drivers and actuator interfaces. The

P. Stone, T. Balch, and G. Kraetzschmar (Eds.): RoboCup 2000, LNAI 2019, pp. 579-582, 2001.
© Springer-Verlag Berlin Heidelberg 2001

communication between the PC and the micro-controllers is via CAN bus and between the PC and other robots or a remote-logging PC is via WaveLAN.

4 Software Architecture

Our approach to robot programming is based on Dual Dynamics (DD) [1], a mathematical model for robot behaviors which we developed. It integrates central aspects of a behavior-based approach, robust control, and a dynamical systems representation of actions and goals. Robot behaviors are specified by differential equations. In the whole they build a dynamical system consisting of subsystems which interact through specific coupling and bifurcation-induction mechanisms. Behaviors are organized in levels where higher levels have a larger time scale than lower levels. Since the activation of behaviors (activation dynamics) is separated from their actuator control laws (target dynamics), we named our approach "Dual Dynamics". An important feature of DD is that it allows for robust and smooth changes between different behavior modes, which results in very reactive, fast and natural motions of the robots.

5 World Model

Dual dynamics behavior systems use symbolic sensors to represent the locally percepted environment of the robot. We do not maintain a global world model shared by all robots. Self localisation of the robot is performed based on odometry and gyroscope data. Since these data is noisy and subject to be disturbed, we compensate odometry errors by improving the self-localization of our robots using *weighted* Monte Carlo sampling [2]. This approach re-adjusts the pose of the robot using the bearing of the goals measured by our vision system.

6 Communication

Our team communication mechanism allows to establish real-time point-to-point connections between all robots of a team. During robot software development all behaviors are kept in a common central source code repository (CVS). Thus we can determine the data flow network within a team of robots in advance. This allows to communicate shared variables between the robots efficiently without any protocol overhead. The variables shared among different behaviour systems are simply tagged in the specification tool DD-Designer. The technical implementation of the team communication mechanism is based on TCP/IP and uses WaveLAN.

7 Skills

Our vision system relies on the well-known Newton Lab's Cognachrome system for ball and goal detection. Since it is mounted on a 360 degree panning unit, we are able to perform "radar-like" optical scans of the robots surrounding. The angle encoder of the panning unit delivers a precise relative angle of each camera picture, which is used in the target dynamics of our behaviors.

Kicking is done with a pneumatic device which is mechanically integrated in the ball guidance of the robot. Kicking is triggered by the behavior system dependent on the pose and the mode of the robot. A neural network is used to anticipate whether the ball will be lost in near future.

The goalie has a slightly different sensor configuration (infrared range detectors pointing to the back of the goal) and of course a specific behavior system. It is essentially a two-dimensional controller, which maintains a fixed distance to the back of the goal and a certain angle to the visible ball. If the robot should be hit by opponent robots thus losing its position in front of the goal, a homing behavior is activated in order to recover the correct position in front of the goal and to re-start the keep goal behavior.

8 Special Team Features

The successful design of robot software requires means to specify, implement and simulate as well as to run and debug the robot software in real-time on a team of physical robots. The integrated Dual Dynamics design environment [2][3] we develop allows to specify DD-models on a high-level of abstraction and to synthesize all code artifacts required to make these models operative in practice: a simulation model, a control program for a real robot, a team communication layer and set-up parameters for real-time monitoring and tracing. In this environment the following steps are performed iteratively in order to design a behavior system for a robot.

Specify. The specification and code generation tool *DD-Designer* comprises a graphical editor to enter the specification of a DD-model in terms of sensors, actors, sensor filters and behaviors. Sensor filters and behaviors are further detailed using the equation editor of DD-Designer. We use multi-target code generation to refine the DD-model to code artifacts required by the simulator, the robot and the real-time monitoring tool. DD-Designer continuously evolved from a first shot prototype [4] to a full-fledged design tool. This development process was an ideal test case to investigate the evolution of software prototypes in design environments [5].

Simulate. The Java simulator *DDSim* is specifically tailored to simulate a team of robots with different behavior systems. The simulator is capable of simulating the whole sensor equipment of the robots including the vision system. The implementation of the behavior system, i.e. the robot control program, is a Java class generated by DD-Designer.

Run. The code for the real robot implements the behavior system in C/C++. This code is directly derived from the high-level specification edited in DD-Designer. Since both artifacts, simulation model (Java) and robot control program (C++), are derived from the same specification, we avoid all problems that occur if a migration from a simulation model to a robot control program has to be performed manually.

Test/Analyse. The real-time trace tool *beTee* allows to capture and analyse internal variable states of an implemented behavior system in real-time [6]. The configuration of beTee is carried out by a Java class generated by DD-Designer. It contains a dictionary of all variables including their dependencies.

9 Future Work

Future work will further focus on our main research goals.

In order to further increase the speed of our mobile robots, the sensor information flow needs to be raised. This will be achieved by adding optical flow sensors to the robot. We will investigate these sensors in close connection with the behavior systems of the robot in order to make them more reactive even at higher speeds than that we already achieved. In addition, we focus on extensions to the Dual Dynamics scheme and on "Observable Operator Models" [7]. Both emphasize the dynamical systems nature of behaviors, with OOMs additionally capturing the stochastic nature of a robot's experience and acting.

In order to further increase the speed of our behavior development process, we have to further narrow the still existing gap between simulation and the real robot. At present, we are able to simulate the robot with its complete sensor equipment in our simulator DDSim. Therefore, we are able to design functional correct behavior systems using simulation only. Nevertheless, parameter tuning is left to real experiments with real robots on the field. We will investigate approaches allowing to adapt the sensor models in the simulator to the precise sensor behavior as measured on individual robots. This will further decrease the number of time-consuming experiments with the robots on the field.

In 2001, it is planned to participate in the 1st GermanOpen in Paderborn and in the 5th RoboCup World Championship in Seattle. Results achieved up to these events will be demonstrated on the GMD-robots.

References

[1] H. Jaeger and Th. Christaller. 'Dual dynamics: Designing behavior systems for autonomous robots', *Artificial Life and Robotics*, 2:108-112, 1998

[2] A. Bredenfeld, T. Christaller, H. Jaeger, H.-U. Kobialka, P. Schoell, 'Robot Behavior Design Using Dual Dynamics', *GMD Report 117,* accepted for KI-Zeitschrift, 2000

[3] A. Bredenfeld, T. Christaller, W. Goehring, H. Guenther; H. Jaeger, H.-U. Kobialka, P. Ploeger, P. Schoell, A. Siegberg, A. Streit, C. Verbeek, J. Wilberg, 'Behavior engineering with "dual dynamics" models and design tools', in *Veloso, M.; Pagello, E.; Kitano, H., (Eds.): RoboCup-99: Robot Soccer World Cup III*, Springer, LNCS 1856, 2000

[4] A. Bredenfeld, 'Co-design tool construction using APICES', in *Proc. of the 7th IEEE Int. Workshop on Hardware/Software Co-Design (CODES'99)*, 1999.

[5] A. Bredenfeld, 'Integration and Evolution of Model-Based Prototypes', *Proc. of the 11th IEEE International Workshop on Rapid System Prototyping (RSP 2000)*, Paris, France, June 21-23, 2000

[6] H.-U. Kobialka and P. Schoell, 'Quality Management for Mobile Robot Development', *6th International Conference on Intelligent Autonomous Systems (IAS-6)*, Venice, Italy, July 25-27, 2000

[7] H. Jaeger, 'Observable operator models for discrete stochastic time series', *Neural Computation* 12(6), 1371-1398, 2000