1. RFC Stuttgart Team Description 2010

U.-P. Käppeler, O. Zweigle, H. Rajaie, K. Häussermann, A. Tamke, A. Koch, B. Eckstein, F. Aichele, D. DiMarco, A. Berthelot, T. Walter and P. Levi

IPVS, University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany
robocup@informatik.uni-stuttgart.de

Abstract. The 1. RFC Stuttgart robot soccer team is used as a testbed for multi-agent software architecture principles in dynamic real time domains. The current research activities focus on a system for advanced data logging using augmented reality and automated camera guidance. It includes approaches from the field of real-time robot message logging, situation based camera control and 3D video visualisation. Another research focus is a genetic algorithm for optimizing the motorcontrol which enables our robots to adapt to different floor and carpet conditions.

1 Introduction

In this work an approach for advanced data logging in the RoboCup domain is presented which includes real-time robot message logging linked to video recording and situation based camera control in combination with 3D rendering algorithms. The RoboCup domain has become an important testbed for robotic applications during the last years. The department Image Understanding of the IPVS - University of Stuttgart is participating with its team 1. RFC Stuttgart in the robotic soccer middle-size league. In this league a team of five autonomous robots is playing together. The robots share and merge information in order to show real team play and cooperative actions. As the robots are communicating data over a wireless network there is the possibility to log, visualize and analyze that data during matches for analyzing for debug purposes. Due to the rapidly growing complexity of the robots software system, which acts in the real world, it becomes more difficult to debug the system in real-time. As a consequence powerful tools for collecting and analyzing data are very important. Those tools have to work in real-time during a match as well as in the post game analysis.

The second part of this team description paper describes an automated configuration for motor controllers. To optimize the low-level drive-behaviour of the robot to get best driving performance we are developing a genetic algorithm approach for optimization of the motor controller parameters of a robot. The goal is an automatic learning approach to obtain the best parameters for the controller without any manual override.

2 Related Work

As the autonomous camera man includes techniques from a lot of different fields, we want to give a short overview over some articles which mainly influenced the development. The information that is exchanged between the robots to generate a common world model [BKL03] is merged to reduce noise in the measurements as described in [BKZ$^+$08]. The team behavior of the cooperative robot system is a result of negotiations, based on a multi agent system (MAS) [Mus00] and can be compared to [Tam97]. The behaviour can be configured by XABSL [LBBJ03] and uses methods like selection equations and interaction nets that are described in [LSZ$^+$07] and [ZLB$^+$06]. The *autonomous camera man* enables an analysis of the team behavior by visualizing the information that is exchanged between the robots and which leads to the decisions in the teams strategy. The basic approach for the camera location determination problem is based on the work of Bolles and Fischler [FB81].

The second part of this team description paper deals with a concept for an automated configuration for motor controllers. To optimize the low-level drive-behaviour of the robot to get best driving performance we are developing a genetic algorithm approach for optimization of the motor controller parameters of a robot. The goal is an automatic learning approach to obtain the best parameters for the controller without any manual override. As shown in several approaches [Gaw86], [MP00], [MP00], [BCP01] self-tuning PID controllers have been developed for several systems using several different methods. Gawthrop [Gaw86] showed an approach of a hybrid PID self-tuner by combination of a continuous time process model and a discrete-time self-tuning controller. Huang and Lam [HL97] presents a method based on genetic algorithms for automatic tuning of proportional, integral and derivative (PID) controllers in Heating Ventilating and Air Conditioning (HVAC) systems to achieve optimal performance. Genetic algorithms, which can considered as search methods based on the theory of Darwin's natural selection, are used in finding near-optimal solutions in complex problem spaces since they have been proved to be robust and efficient. Thus our intention was to improve our robots driving performance using these methods to find the best PID parameters for our motors according to the environment.

3 Debugging in a Multi-Agent Robot System

In order to improve the general logging capabilities of robot systems and to visualize the corresponding data in a way that is easy to understand for humans we implemented an *autonomous camera man* and a visualization using augmented reality. The system consists of a camera, a pan-tilt unit (PTU) and a software for controlling both, which is connected to the team's communication channel which our robots are using during a match, to automatically steer the pan-tilt unit where the camera is mounted on. As a consequence a whole match can be filmed autonomously without human interference. Furthermore the system is rendering all the information gained from the robots into the live image of the camera. This allows for interpreting all scenes in a much better way. Moreover it is possible to save the whole video stream and data stream from the robots. Consequently the data and video stream can be replayed and even the kind and amount of information rendered into the video stream can be changed and adapted. Such a system is based on different approaches from autonomous systems, 3D visualization and vision.

3.1 Automated camera guidance

The PTU is controlled via a serial device, which makes it impossible to steer the device in real-time. The only information available consists of the commands that have been sent to the device. In order to merge the coordinate systems of the video and augmented scene, we have to know precisely the current orientation of the camera. To solve that problem an observer tracks the angles of the PTU.

The coordinates of the object of interest are first chosen due to the current state of the shared world model. Those are transformed to the camera frame and the corresponding pan- and tilt angles are calculated. They are sent to both the device and the observer. The visualization process always uses the values of the observer. Correction of the estimated position of the observer is done via *"bang-bang control"*, because the only fact known when reading the values from the device is, whether the observer has already been in position or not. The step of adjustment can be chosen by the time difference between PTU and observer in reaching the wanted position. Experience showed that the acceleration phase of the stepper engines is negligible small. So the observer equation can be obtained as

$$\theta_{n+1} = \theta_n + (\theta_n - \theta_C)(t_{n+1} - t_n)v_C \delta v_e \delta v_e = t_f^{obs} - t_f^{pt} \quad (1)$$

where θ_n is the pan-, resp. tilt-, angle internally calculated in the observer, variables with subscript. C are the values of the latest command that has been sent to the real PTU.

3.2 Augmented reality

The grabbed image from the camera is directly mapped on a GL texture, This enables an overlay with information from the robots world models and any 3D mapping of the image in our Augmented Reality Scene. Therefore the field of vision is not limited to the view of the camera, as shown in Figure 1. It can be extended to include more information that is exchanged between the robots but not included in the field of vision of the camera. This extends the possibilities of analyzing recorded scenes.

Objects of interest All information for the augmented scene is gathered via the communication of the robots using a message dispatch server. The robot control graphical user interface (RCG) holds a copy of all transfered data. Usually the ball is used as an indicator for the most important view frustrum. But in some

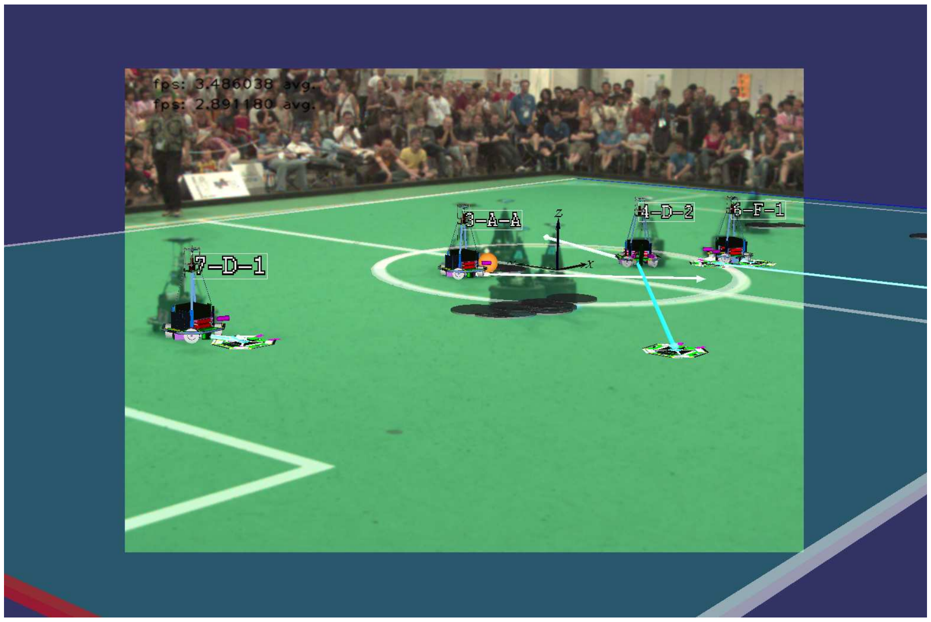


Fig. 1. Screenshot of the videobox in the scene, when the view matrix is fixed to the estimated camera position.

cases another object might be of interest, e.g. when the ball is shot with high velocity so that it is impossible for the robots to track it. If our team tries to score, the opponent's goal is chosen as center for the camera direction. Our goalkeeper is centered when the opponent is attacking our goal.

3.3 Camera Pose Framework

In order to overlay the abstract information with the video stream a basic necessity is to find the camera position with respect to the RoboCup game field.

Camera location determination The Location Determination Problem is a common problem in image analysis and can be formulated as follows [FB81] : "Given a set of m control points, whose 3-dimensional coordinates are known in some coordinate frame, and given an image in which some subset of the m control points is visible, determine the location (relative to the coordinate system of the control points) from which the image was obtained".

A complete solution to this problem is described by Bolles and Fischler [FB81]. The problem is simplified into having a tetrahedron, whose three base's vertices coordinates are known (3 control points), and where each angle to any pair of the base's vertices from top are also known. These angles are computed using the image properties and the location of the control points on this image. Knowing these values, it is possible to solve a biquadratic equation which can give up to four possible solutions.

The camera is located near to the field, which makes it impossible to obtain a sufficient amount of control points on a single image. Therefor a different way for obtaining the needed angles has been found. The camera is attached to a PTU, whose coordinates frame is not known in the soccer field's frame. We know the *3D*-coordinates of the control points, but it's impossible to determine the necessary angles from a single picture. As the camera is attached to the PTU, the idea is to focus the control points step by step, and to measure the angles between the origin position (also called *0*-position) and the position heading to each control point as shown in Figure 2(a) and 2(b). Afterwards the camera location can be determined as described by Bolles and Fischler [FB81].

In order to compute the angles offset of the camera, a new approach is used which differs from the one described by Bolles and Fischler [FB81]. This new approach consists of comparing the angles used by the pan-tilt unit to focus the control points, and the theoretical angles it should have used if there was no angle offset. There are three types of angle offsets for the camera : *pan*, *tilt* and *pitch*. For comparing those values it is essential not to give importance to the geometrical meaning of the angles. Just consider them as a set of numerical data which properties have to be determined and compared. The angles previously measured are *pan* und *tilt*, and we consider these angles as *2*-dimensional points with *pan* as *x*-coordinate and *tilt* as *y*-coordinate, so that the pan offset and tilt offset are equivalent to a translation (*Pan* and *Tilt* are the only two angles which are driven by the PTU). The pitch offset is equivalent to a rotation with the origin of the frame as center of rotation (the Pan Tilt unit is mounted in a way that the *0*-position is also the axis of rotation for *pitch*).

The pitch offset is computed with a Principal Component Analysis (PCA): it's the angle between the two principal components. Then the pitch offset has to be compensated for computing the other offsets, rotating the measured values by the previously computed pitch offset. The pan offset and the tilt offset can be derived simply by translating between the two sets of values (theoretical and compensated) as depicted in Figure 3.

3.4 Analysis and Export

During a robocup match our automatic cameraman can record a video and the communication between the robots. Afterwards it is possible to replay the video and to include the overlays generated from the messages of all robots. The possibility of modifying the speed of the replay or to display each image of the sequence step by step enables a detailed analysis of the robot's interpretation of sensor data. Changes in a robot's role or strategy can be compared to Positions of obstacles in the robot's world model and to the real world at the same time.

An export of a video including the camera images and all overlays enables an exchange of the logs between the teams and a simple playback of a robocup game on any pc.

4 Self-learning motorcontrol

The motor controller [Max10] of the robot has a set of many parameters that can be changed in order to achieve the best driving-performance according to the environment [Lip06]. To evaluate such a set of parameters, it is necessary to run a short test drive that takes several seconds. With billions of possible paramenter combinations, a simple try-and-error method will take far too long to reach a solution.Thus we are forced to use methods from the machine learning theory. Beside methods of reinforcement learning another promising approach to solve this problem is the use of genetic algorithms [Gol89],[JJdSC$^+$03], [ES03],[MYK99], [Gaw86]. Genetic algorithms, which can be considered as search methods based on the theory of Darwin's natural selection, are used since they have been proved to be robust and efficient in finding near-optimal solutions in complex problem spaces. The interdependence of some parameters of the motor controller and the effect to the driving behavior is shown in figure 4.

4.1 Genetic algorithms (GA)

The core part of the genetic algorithm is its fitness function where the test drive is performed. This makes it the most time consuming part of the optimization process - therefore one of the main challenges is to minimize the number of calls of the fitness function. For example it can be an advantage to let the parents survive if they have a higher fitness than their children. On the other hand, this can be a problem if the population is very small and the genetic variance becomes too low [ES03]. To prevent this, it seems reasonable to use all individuals for the recombination and to make the selection only by comparing the fitness of the parents with their children. At that point, we have to decide whether comparing only the children with their own parents or sort the fitness of all individuals and take the best, independent from which generation they are. Another interesting approach is to switch after a few generations to evolution strategies. This is useful if we are close to the global maximum and want to exploit it.

4.2 Fitness function

The most difficult part is to find a good fitness function with some criteria to determine how well a solution performs. Therefore we give a certain desired speed directly to the low-level controller and measure the effective speed of the wheels. The goal is to minimize the difference between these two values. For evaluation purposes we create a test-run with different speeds, unit steps, ramps and so on. To calculate the summarized error of one complete test-run we are evaluating different methods (e.g. quadratic mean). The only limiting factor is the time we need for such a run. Another problem is the uncertainty and the slippage due to the wheels during the acceleration in the test-runs that results in a change of the direction of the robot. This requires a permanent check of the current position which can be also one criteria for the fitness-function. The state of the battery charge is also a variable we have to check for each run, because a low voltage could be results in negative behavior changes. The mathematical description of one of our fitness-functions is shown in the following:

$$\alpha(t+1) = \alpha(t) + (abs(v_{set} - v_{real})^{\lambda} \tag{2}$$

$$fitness(t) = \frac{\gamma}{(\alpha(t) - \alpha_{min})} \tag{3}$$

Where γ, α_{min} and λ equates to a learning-rate parameter. v_{set} corresponds to the angular-velocity set by the controller to the motor and v_{real} corresponds to the real values (measured by odometry).

5 Conclusion and Outlook

In this paper we presented an approach for a self-learning configuration of motor controller and an autonomous camera man as a powerful data logging tool. Practical experience showed that the approach for logging game data in a RoboCup match makes it much easier for the developers to debug robot software using a 3D visualization which is easy to interpret.As a further step the system should use the panorama camera LadyBug II, which is able to create 360degree images. Transfering the camera location determination problem to such a system will be a new challenge.

References

[BCP01] R. Bandyopadhyay, UK Chakraborty, and D. Patranabis. Autotuning a PID controller: A fuzzy-genetic approach. *Journal of Systems Architecture*, 47(7):663–673, 2001.

[BKL03] T. Buchheim, G. Kindermann, and R. Lafrenz. A dynamic environment modelling framework for selective attention. In *IJCAI Workshop: Issues in Designing Physical Agents for Dynamic Real-Time Environments: World modeling, planning, learning, and communicating.* IJCAI, 2003.

[BKZ$^+$08] Ruben Benkmann, Uwe-Philipp Käppeler, Oliver Zweigle, Reinhard Lafrenz, and Paul Levi. Resolving Inconsistencies using Multi-agent Sensor Systems. In Luis Seabra Lopez, Filipe Silva, and Vitor Santos, editors, *Proceedings of the 8th Conference on Autonomous Robot Systems and Competition: Robotica 08; Aveiro, Portugal, April 2nd, 2008*, pages 93–98, Aveiro, April 2008. Universidade de Aveiro.

[ES03] A.E. Eiben and J.E. Smith. *Introduction to evolutionary computing.* Springer Verlag, 2003.

[FB81] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[Gaw86] P. Gawthrop. Self-tuning PID controllers: Algorithms and implementation. *IEEE Transactions on Automatic Control*, 31(3):201–209, 1986.

[Gol89] D.E. Goldberg. *Genetic Algorithms in Search and Optimization.* Addison-wesley, 1989.

[HL97] W. Huang and HN Lam. Using genetic algorithms to optimize controller parameters for HVAC systems. *Energy & Buildings*, 26(3):277–282, 1997.

[JJdSC$^+$03] M. Jamshidi, M. Jamshidi, L. dos Santos Coelho, R.A. Krohling, and P.J. Fleming. *Robust control systems with genetic algorithms.* CRC, 2003.

[LBBJ03] M. Lötzsch, J. Bach, H.-D. Burkhard, and M. Jüngel. Designing agent behavior with the extensible agent behavior specification language xabsl. In *7th International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences*, 2003.

[Lip06] B.G. Liptak. *Instrument Engineers' Handbook: Process control and optimization.* CRC, 2006.

[LSZ$^+$07] Reinhard Lafrenz, Frank Schreiber, Oliver Zweigle, Michael Schanz, Hamid Rajaie, Uwe-Philipp Käppeler, Paul Levi, and Jens Starke. Evaluating coupled selection equations for dynamic task assignment using a behavior framework. *Berns, K. (ed.); Luksch, T. (ed.): Autonome Mobile Systeme 2007*, 1(1), 2007.

[Max08] Maxon Motor. Encoder HEDL 5540 Data sheet. `test.maxonmotor.com/docsx/Download/catalog_2005/Pdf/05_246_e.pdf/`, November 2008.

[Max10] Maxon Motor. EPOS 70/10 Data sheet. `test.maxonmotor.com/docsx/Download/catalog_2005/Pdf/05_271_e.pdf/`, January 2010.

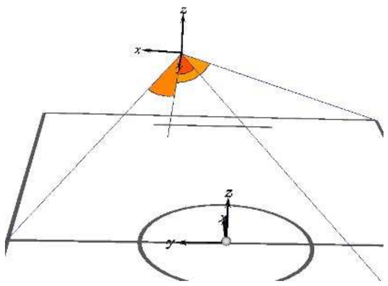
[MP00] R.K. Mudi and N.R. Pal. A self-tuning fuzzy PI controller. *Fuzzy Sets and Systems*, 115(2):327–338, 2000.

[Mus00] K.M. Muscholl. *Interaktion und Koordination in Multiagentensystemen.* Dissertation, Universität Stuttgart, 2000.

[MYK99] Y. Mitsukura, T. Yamamoto, and M. Kaneda. A design of self-tuning PID controllers using a genetic algorithm. In *American Control Conference, 1999. Proceedings of the 1999*, volume 2, 1999.

[Tam97] M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.

[ZLB$^+$06] Oliver Zweigle, Reinhard Lafrenz, Thorsten Buchheim, Hamid Rajaie, Frank Schreiber, and Paul Levi. Cooperative agent behavior based on special interaction nets. In *Intelligent Autonomous Systems 9*, 2006.


(a) Angles from *0*-position up to control points are measured.

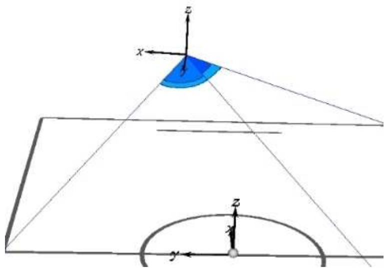

(b) Angles from center of perspective to any pair of control points are computed, using previously measured angles.

Fig. 2. Compute angles for Bolles and Fischler's method.

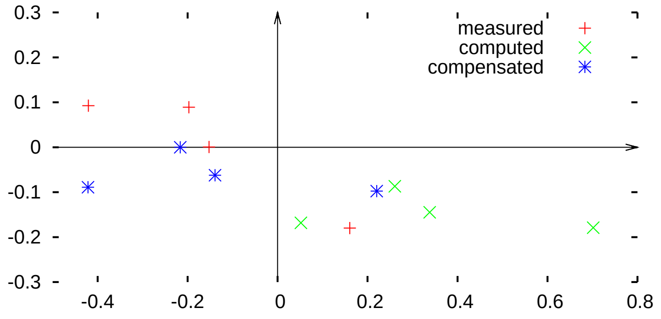


Fig. 3. Example of computed angle offsets. With the pitchoffset compensation, the other offsets are just the translation coordinates.

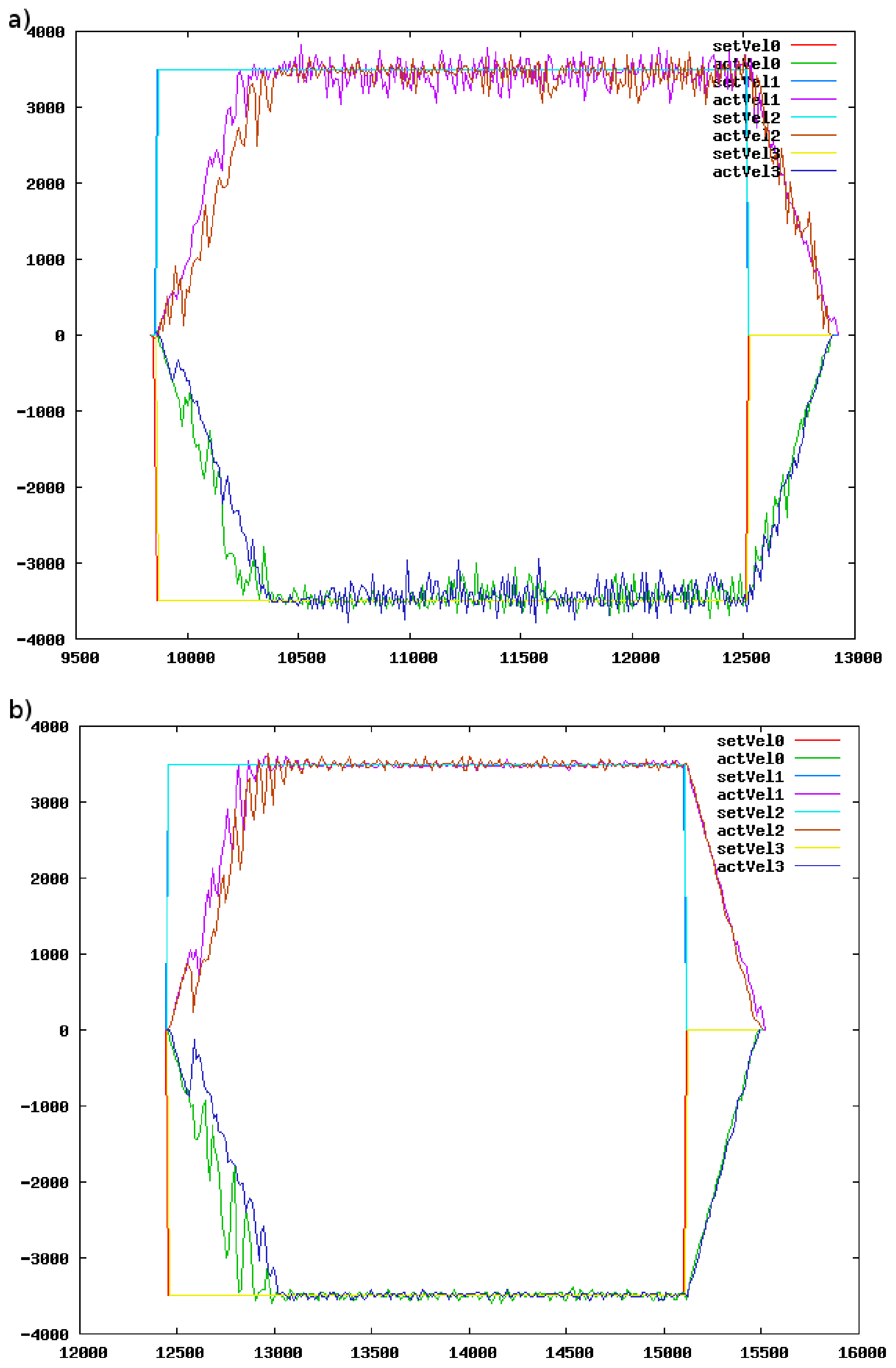


Fig. 4. The diagramm shows the difference between the angular velocity set to the motors and the real values (measured by encoders [Max08]). a) poor PID-Parameters (delta between set and real values is high) b) optimal PID-Parameters (delta between set and real values is low)