A Behaviour-Based Control and Software Architecture for the Visually Guided Robudem Outdoor Mobile Robot

Daniela Doroftei, Eric Colon, Geert De Cubber

Abstract:

The design of outdoor autonomous robots requires the careful consideration and integration of multiple aspects: sensors and sensor data fusion, design of a control and software architecture, design of a path planning algorithm and robot control. This paper describes partial aspects of this research work, which is aimed at developing a semi-autonomous outdoor robot for risky interventions. This paper focuses on three main aspects of the design process: visual sensing using stereo vision and image motion analysis, design of a behaviourbased control architecture and implementation of modular software architecture.

Keywords: *???.*

1. Introduction

For decades, autonomous robotics has been popular research area, yet the amount of real intelligent autonomous outdoor robots applied on the field is still very limited. The research project presented here has as a goal to develop intelligent autonomous robotic agents which can assist humans for various types of risky outdoor interventions (surveillance, crisis management, ...). The challenges for such a robotic system are tremendous and span various fields of research: from sensing and sensor fusion to modelling and control, map building and path planning, decision making and autonomy, to the final integration of all these components. Three of these aspects are investigated more profoundly in this paper: visual sensing, robot control and software architecture. The following paragraphs give an overview of the different existing algorithms and design choices for these different components.

Robotic agents can rely on numerous types of sensors to gain knowledge about the environment or about itself and its position. These sensors include infrared sensors, ultrasound sensors, laser range scanners, GPS, inertial measurement units, ... Cognitive science and biological examples pointed out the importance of visual sensing, which led to the application of computer vision algorithms like stereo vision [12], obstacle detection [6], person following [8], visual servicing[10] to robotics and it eventually also led to mixed paradigms like visual simultaneous localisation and mapping (VSLAM) [4]. However, integration of vision modules into control architecture for an autonomous mobile robot is more difficult than just adding the vision components. [17] This is due to the high bandwidth and processing requirements of vision sensors, which require a task-specific configuration of vision-based behaviours. Another drawback of many computer vision algorithms is the lack of stability and robustness when confronted with varying illumination conditions, as in outdoor situations, although illumination-invariant algorithms have been proposed [5]. For visual sensing, the Robudem robot is outfitted with a stereo camera system, consisting of two digital cameras. In order to maximize the information stream towards the navigation unit, two different visual processing techniques are used: stereo vision and image motion analysis. An autonomous mobile robot must be self-reliant to operate in complex, partially known and challenging environments using its limited physical and computational resources. Its control system must ensure in real time that the robot will achieve its tasks despite all these constraints. [11] One of the first robot control architectures was the Sense Model Plan Act (SMPA) paradigm. The primary drawback of this approach is that the series of stages through all sensor data must pass places an unavoidable delay in the loop between sensing and action. To counter this drawback, alternatives, such as the behaviour-based approach, were proposed. In behaviour-based control, the control of a robot is shared with a set of purposive perception-action units, called *behaviours*. [14] Based on selective sensory information, each behaviour produces immediate *reactions* to control the robot with respect to a particular objective, a narrow aspect of the robot's overall task such as obstacle avoidance or wall following. Behaviours with different and possibly incommensurable objectives may produce conflicting actions that are seemingly irreconcilable. Thus a major issue in the design of behaviour-based control systems is the formulation of effective mechanisms for coordination of the behaviours' activities into strategies for rational and coherent behaviour. This is known as the *action selection problem*. Numerous action selection mechanisms have been proposed over the last decade; a qualitative overview can be found in [7]. The behaviour-based controller presented here uses of statistical reasoning on the output data of each behaviour to determine the stability and reliability and therefore also the activity level of seven behaviours, each proposing a (different) velocity and turning setup value for the robot actuators.

Today, robot control architectures become more and more complex, as human reasoning is mimicked. Moreover, there is a significant portion of robot functionality that is common to a large number of robotic systems in different application domains. Unfortunately, most functionality implementations are tied to specific robot hardware, processing platforms, and communication environments. Most research and development in software for robotic systems is based on proprietarily designed architectures invented from scratch each time. To avoid this problem, the choice of flexible, extendable and real-time capable software architecture is very important. This software architecture has to ease the use of reusable and transferable software components. Multiple software architectures, like Orocos [2], Player/Stage [9], Campout [15], CoRoBa [3] … have been proposed in the past, all with their strengths and weaknesses. The existence of such a multitude of software frameworks hasn't helped the standardisation of robot software architectures. In the course of this research project, the Modular Controller Architecture (MCA) [18] was used. MCA is a modular, network transparent and realtime capable framework tailored to the control of autonomous robots.

The remainder of this paper is structured as follows: first the visual sensing algorithms are explained in detail; then a behaviour based robot control scheme is proposed after which the implemented software architecture is introduced.

2. Visual sensing

2.1. Stereo Vision

Stereo vision employs the difference in location between two cameras. This difference in space leads to two images where the same points can be found at different positions. The goal of stereo disparity estimation is finding the correct correspondences between image points from the left and right camera. For this, we employ the algorithm presented by Birchfield *et al.* in [1]. The algorithm matches individual pixels in corresponding scan line pairs while allowing occluded pixels to remain unmatched, and then propagates the information between scan lines. The algorithm handles large untextured regions, uses a measure of pixel dissimilarity that is insensitive to image sampling, and prunes bad search nodes to increase the speed of dynamic programming. The output of this algorithm is a dense depth map of the area in front of the cameras, as shown in Figure 1. On the depth map in Figure 1, nearby objects appear dark. The cross on top marks the location of the closest obstacle, which is the darkest point on the depth map and which corresponds here to the obstacle in front of the robot.

The data-content of this dense depth map must now be reduced to be useful for the navigation controller. For this, we use the approach proposed by Schafer in [16]. Following this approach, the dense depth map is down projected onto the ground surface, such that it can be represented as a 2D line. This data is further reduced in dimensionality by calculating from the depth line the distance to the nearest obstacle on the left d_l , in the middle d_c, and on the right d_r , respectively.

2.2. Image Motion Analysis

Motion analysis can provide extra information about the environment. The rationale behind the usage of the image motion for navigation purposes is that when large image motion is detected, this is likely due to objects close to the camera (and thus close to the robot), which should trigger the obstacle avoidance module. On the other hand, when few image motions are detected, this means that the way in front of the camera is probably quite clear of obstacles.

Multiple techniques stand at our disposal to estimate the image motion. These methods differ in their approach towards the main problem to be solved in image motion: the background estimation and subtraction process. As the camera system is installed on a moving robot system, background estimation is particularly difficult in this case, as it is very hard to build up a model of the background over a large amount of time. This constraint the use limits of traditional advanced background estimation techniques like kernel density estimators, mean shift or mixtures of Gaussians [13]. As a result, the frame difference between successive frames was employed to find back the moving objects. As expressed by equation (1), the motion m_k for each pixel is robustly estimated by calculating the frame difference when the difference is above a certain threshold which is dependent on the robot velocity V.

$$\mathrm{m_k} = \begin{cases} 0 & \text{if } \left|\mathrm{frame_i} - \mathrm{frame_{i-1}}\right| < \mathrm{cV} \\ \left|\mathrm{frame_i} - \mathrm{frame_{i-1}}\right| & \text{if } \left|\mathrm{frame_i} - \mathrm{frame_{i-1}}\right| > \mathrm{cV} \end{cases} \tag{1}$$

With c a constant describing the relation between robot speed and image motion. Figure 2. shows this image motion field as calculated by the right robot camera.

The resulting motion field m is then summed over the whole image to obtain one single motion measure for the camera image.

$$\mathrm{m} = \sum_{\mathrm{k}=1}^{\text{all pixels}} \mathrm{m_k} \tag{2}$$

This calculation is performed once for the left camera and once for the right camera image, leading to two distinct image motion estimates.

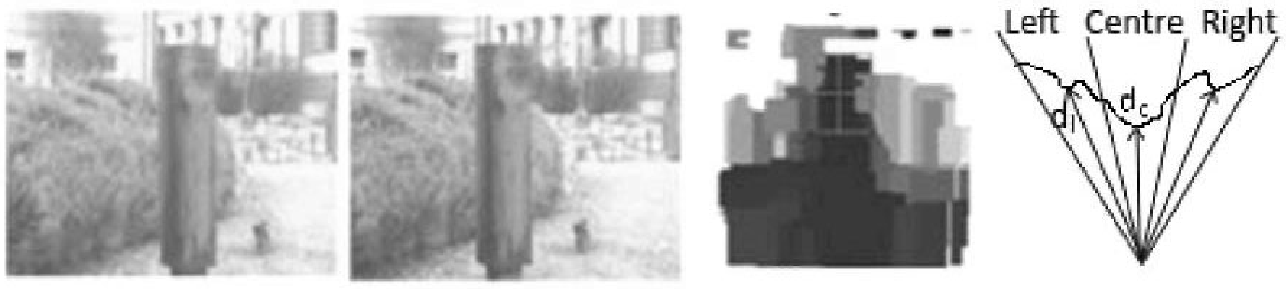

Fig. 1. Stereo Processing a) Left camera image; b) Right camera image; c) Dense depth map (white=far, dark = near); d) Depth line with nearest distances to obstacles on the left, in the middle and on the right.

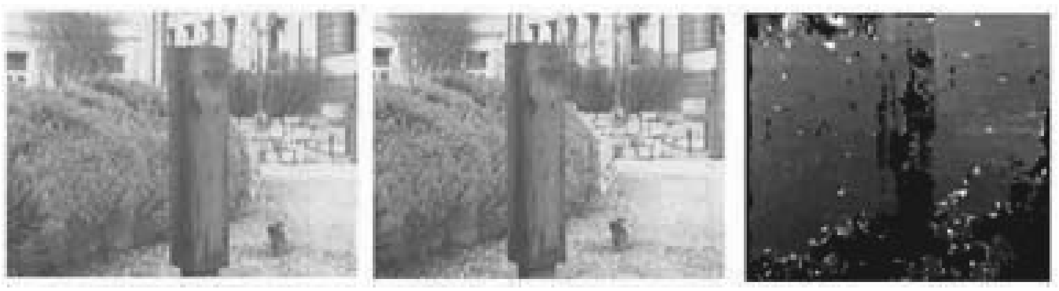

Figure 2. Image motion field of right camera: a) Image at t_{i-1}; b) Image at t_i; c) Image motion field.

3. Behaviour-based robot control

3.1. Behaviour-based framework

The control architecture describes the strategy to combine the three main capabilities of an intelligent mobile agent: sensing, reasoning and actuation. These three capabilities have to be integrated in a coherent framework in order for the mobile agent to perform a certain task adequately. To combine the advantages of purely reactive and planner-based approaches, this research work aims at implementing a behaviour-based controller for autonomous navigation.

For the Robudem robot, three main sensing capabilities are installed: odometry, stereo vision, image motion analysis. These three sensing capabilities are processed by separate behaviours. In this context, the odometry information is used for guiding the robot to a given goal position, while the visual sensors are used for obstacle avoidance. The main advantage of using behaviour-based approaches is that each of the different behaviours can be designed and implemented separately. For the Robudem robot, three main sensing capabilities are installed for now: odometry, stereo vision, image motion analysis. These three sensing capabilities are processed by separate behaviours. In this context, the odometry information is used for guiding the robot to a given goal position. The general scheme of the behaviour-based navigation module is shown in Figure 3.

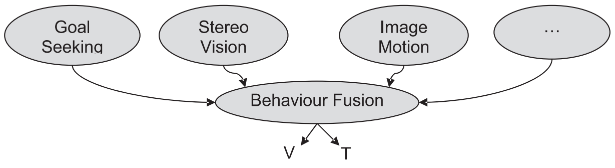

Fig. 3. General scheme of the behaviour-based navigation module.

The output of this behaviour-based navigation planner is a velocity V and turn angle T command to be sent to the robot. How these properties are estimated is explained in detail in the following section.

3.2. Behaviour design and fusion

The velocity V and turn angle T sent to the robot are dependent on the individual input of each of the different behaviours: goal seeking, stereo vision and image motion. Therefore, each of these behaviours calculates its own velocity and turn angle for the robot to be performed. These values are then weighed according to the activity level A of the specific behaviour according to the formulation of equation (3). The activity level of behaviour describes to which degree this behaviour is relevant for the calculation of the final robot command.

$$V = \left(1 - A_E\right)\left(\frac{A_S V_S + A_M V_M + A_P V_P}{A_S + A_M + A_P}\right)$$
$$T = A_{F,R}\left(\frac{A_S T_S + A_M T_M + A_P T_P}{A_S + A_M + A_P}\right) \tag{3}$$

With V and T, respectively, the velocity and turning command for the robot; A_E, A_F, A_R the activity level for, respectively, the Emergency Stop and the Front or Rear Steering behaviour; A_S, A_M, A_P the activity levels for, respectively, the Stereo Vision, Image Motion and Goal Seeking behaviour; V_S, V_M, V_P the Velocity commands from, respectively, the Stereo Vision, Image Motion and Goal Seeking behaviour; T_S, T_M, T_P the turn angle commands from, respectively, the Stereo Vision, Image Motion and Goal Seeking behaviour.

When the Emergency Stop is activated, no robot movement is possible, which is a security measure. The activity level for the front and rear steering decides on the drive mode, which will be adopted by the robot. The velocity and turn prescripts are calculated at behaviour level as follows:

- The stereo vision behaviour receives as data d_l and d_r, the distances to obstacles on the left and right side of the robot. The smaller the distance to obstacles, the more careful and slow the robot must move. The velocity V_S is therefore directly proportional to the mean of the measured distances left and right and the turn angle T_S is chosen to maximize the distance to obstacles.
- The image motion behaviour receives as data m_l and m_r, the movement measured by the left and the right camera. The more movement in the image, the more probable there are objects close to the robot, so the velocity should be lowered. The robot speed V_M is as such inversely proportional to the image motion and the turn angle T_M is chosen to minimize the image motion.
- The goal seeking behaviour receives as data the estimate of the robot position (x_R, y_R) and orientation θ_R as calculated by the odometry and robot kinematics.
- This position and orientation is compared to the desired goal position (x_G, y_G) and orientation θ_G and a velocity V_P and turn angle T_P prescript are calculated from this information.

A major issue in the design of behaviour-based control systems is the formulation of effective mechanisms for coordination of the behaviours' activities into strategies for rational and coherent behaviour. Therefore, the activity levels need to be calculated. As noted before, these activity levels should reflect the relevance of the specific behaviour. The principle behind the calculation of the activity levels is that the output of behaviour should be stable over time in order to trust it. Therefore, the degree of relevance or activity is calculated by observing the history of the output - a velocity and turn angle - of each behaviour. This history-analysis is performed by comparing the current output to a running average of previous outputs, which leads to a standard deviation, which is then normalized. For the stereo vision behaviour, these standard deviations are:

$$\sigma_{S,V} = c_V \sum_{k=i-h}^{i} \left(V_{S,k} - \frac{\sum_{j=1}^{N} V_{S,j}}{N} \right)^2$$
$$\sigma_{S,T} = c_T \sum_{k=i-h}^{i} \left(T_{S,k} - \frac{\sum_{j=1}^{N} T_{S,j}}{N} \right)^2 \quad (4)$$

With c_V and c_T two normalisation constants.

The bigger this standard deviation, the more unstable the output values of the behaviour are, so the less they can be trusted. The same approach is followed for the image motion (subscript M) and the goal seeking (subscript P) behaviours. This leads to an estimate for the activity levels:

$$A_{S/M/P} = \left(1 - \sigma_{S/M/P,V}\right) + \left(1 - \sigma_{S/M/P,T}\right) \quad (5)$$

For stability reasons, the activity level is initialized at a certain value (in general 0.5) and this estimate is then iteratively improved.

4. A modular software architecture

4.1. An introduction to MCA

As control architectures which aim at mimic human thinking risk of becoming highly complex, the choice of a flexible, extendable and real-time capable software architecture is very important. This software architecture has to ease the use of reusable and transferable software components. The chosen software architecture, MCA (Modular Controller Architecture) as presented by Scholl in [18], achieves this by employing simple modules with standardized interfaces. They are connected via data transporting edges which is how the communication between the single parts of the entire controller architecture is managed. The main programs only consist of constructing modules that are connected *via* edges and pooled into a group. This results in an equal programming on all system levels. As modules can be integrated both on Windows, Linux and on RT-Linux without changes, they can be developed on Linux-side and then transferred later to RT-Linux. As errors in RT-Linux lead to system-hangs this development strategy prevents from many reboot cycles and results in faster software development.

Each MCA module has a structure as shown in Figure 4. and is determined by four connectors with the outside world: Sensor input (left below), Sensor output (left top), Control input (right top), Control output (right below). As a result sensor data streams up, control commands stream down. The Sensor input and output are connected through a Sense procedure which enables to process the sensor data and the Control input and output are connected through a Control procedure which enables to process the control commands.

This modular structure is particularly convenient for behaviour-based architectures as the individual behaviours translate easily to corresponding MCA-modules.

4.2. The proposed behaviour-based control architecture

Figure 4. shows the MCA scheme for the behaviour-based controller subgroup. It consists of three main behaviours, controlled (fused) by a Behaviour Selector module.

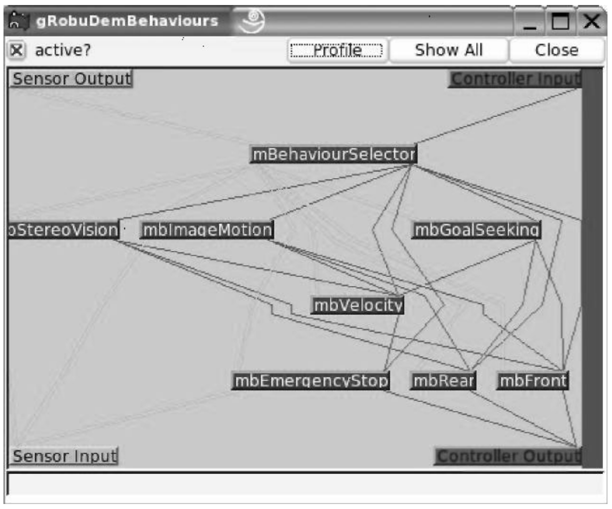

Fig. 4. MCA scheme for the behaviour-based controller subgroup.

The sensory input received by the behaviour-based controller subgroup consists of the distances to obstacles: d_l, d_c, d_r; the motion in the left and right camera m_l, m_r and the robot position (x_R, y_R) and orientation θ_R. This data is processed by the stereo vision, image motion and goal seeking behaviours, outputting a velocity and turning command. The *BehaviourSelector* module receives as input the output of the different behaviours and calculates the activity levels for each of these behaviours according to equations (5). All of the three main behaviours also send the velocity command which was calculated to the Velocity module, where a fusion of the data occurs, using the activity levels, calculated by the *BehaviourSelector* module. The *EmergencyStop* module, which can be triggered by the user or when an obstacle is detected within a security distance, ensures that the velocity command is only transferred to the robot in safe conditions. For the turning behaviour, a similar approach is followed; only in this case there are two separate fusion modules where the navigation behaviours can send their results to. This is related to the mechanical structure of the Robudem robot, which has a two-by-two differential drive system, meaning front and back wheels can be given a different turning angle, allowing for highly flexible manoeuvring on difficult terrain. It is our aim to let the behavioural controller (BehaviourSelector) decide, which is the best drive mode given the terrain circumstances, by setting the activity levels for the *FrontSteering* and *RearSteering* modules. For now, the user can select on the graphical user interface the desired drive mode and the activity levels A_F and A_R are set accordingly.

5. Results

The achievements of this research project can be discussed by taking a look at the Graphical User Interface (GUI) in Figure 5., as it shows the important features of the Robudem controller. On the upper left, the stereo camera images are visible, which are rectified to facilitate the calculation of the depth map. The dense depth map is then post processed, as is shown on the images to the right of the original depth map. In the middle of the interface, the measurements of the abstract visual sensors stereo vision and image motion analysis are shown using coloured bars. These indicate for the stereo vision sensor the distances to obstacles on the left, middle and centre and for the image motion sensor the motion in the left and right camera image.

At the right of the interface, the activity levels of the different behaviours are shown. As can be noticed, the *BehaviourSelector* has found here that the *ImageMotion* behaviour delivers more reliable results than the *StereoVision* behaviour. This is in this case due to the lack of texture in the camera images, which renders the dense depth map estimation less robust. As discussed before, the activity levels for the *VelocitySteering*, *FrontSteering* and *RearSteering* are user-decided by setting the drive mode.

5. Conclusions

In this paper we presented three main aspects of the design process of an intelligent autonomous robotic agent for risky outdoor interventions: visual sensing, behaviour-based control and the software architecture. Multiple visual cues, stereo vision and image motion analysis, were integrated into the robot control and software architecture. Behaviour based control architecture was proposed, using statistical reasoning to solve the action selection problem. All components were implemented using modular software architecture to achieve a future-proof design. The integration of these aspects enables the robot to reach a designated goal while avoiding obstacles.

AUTHORS

Daniela Doroftei*, Eric Colon and Geert De Cubber - Royal Military Academy, Department of Mechanical Engineering (MSTA), Av. de la Renaissance 30, 1000 Brussels. E-mails: {daniela.doroftei, eric.colon, geert.de.cubber} @rma.ac.be

* Corresponding author

References

[1] S. Birchfield, Klt: *An implementation of the kanade-lucas-tomasi feature tracker*, January 1997. Available at: http://www.ces.clemson.edu/stb/klt/.

[2] H. Bruyninckx, "Open robot control software: the OROCOS project", *Proc. Int. Conf. on Robotics and Automation*, vol. 3, 2001, pp. 2523-2528.

[3] E. Colon, H. Sahli, Y. Baudoin, "CoRoBa, a multi mobile robot control and simu-lation framework", *Int. J. of Adv. Robotic Systems*, vol. 3, no. 1, 2006, pp. 73-78.

[4] A.J. Davison, I.D. Reid, "MonoSLAM: Real-Time Single Camera SLAM", *IEEE Trans. on pattern analysis and machine intelligence*, vol. 29, no. 6, June 2007.

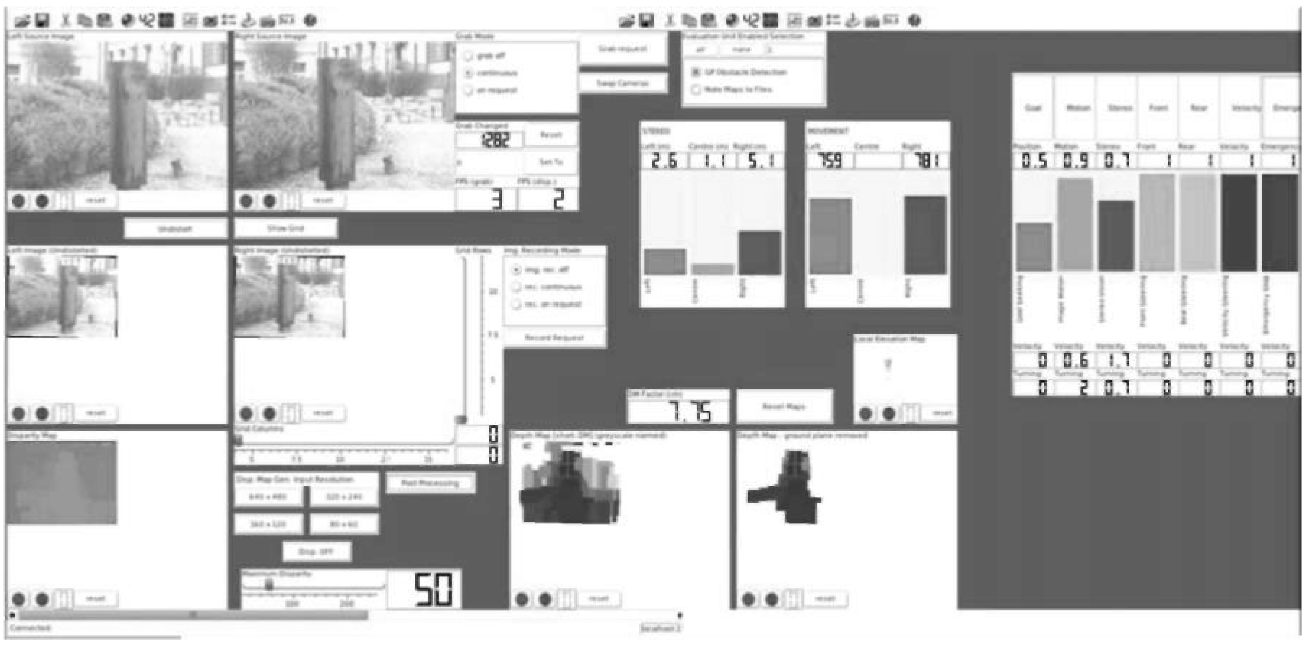

Fig. 5. Graphical User Interface of the Robudem navigation controller.

[5] G. De Cubber, S. Berrabah, H. Sahli, "Color-Based Visual Servoing Under Varying Illumination Conditions", *Robotics and Aut. Systems*, vol. 47, no. 4, 2004, pp. 225-249.

[6] G.N. DeSouza and A.C. Kak, "Vision for Mobile Robot Navigation: A Survey", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 2, Feb. 2002 pp. 237-267.

[7] D. Doroftei, *Behavior based Robot Navigation Techniques - State of the Art*. Tech. report & Lecture Notes of The Royal Military Academy, Belgium, October 2006.

[8] V. Enescu, G. De Cubber, K. Cauwerts, S. Berrabah, H. Sahli, M. Nuttin, "Active Stereo Vision-based Mobile Robot Navigation for Person Tracking", *Int. Conf. on Informatics in Control, Automation and Robotics*, vol. 2, 2005, pp. 32-39.

[9] B. Gerkey, R.T. Vaughan, and A. Howard, "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems". In: *Int. Conf. on Advanced Robotics*, Portugal, 2003.

[10] P. Hong, H. Sahli, E. Colon, Y. Baudoin, "Visual Servoing for Robot Navigation". In: *3rd Int. Conf. on Climbing and Walking Robots*, Germany, 2001, pp. 255 - 264.

[11] A.D. Medeiros, "A survey of control architectures for autonomous mobile robots", *J. Braz. Comp. Soc.*, vol. 4, no. 3, 1998.

[12] C. Park, S. Kim, J. Paik, "Stereo vision-based autonomous mobile robot". In: *Intelligent Robots and Computer Vision XXIII: Algorithms, Techniques, and Active Vision*. Proceedings of the SPIE, vol. 6006, October 2005, pp. 256-264.

[13] M. Piccardi, "Background subtraction techniques: a review". In: *Proc. of the IEEE Int. Conf. on Systems, Man and Cybernetics*, vol. 4, October 2004, pp. 3099-3104.

[14] P. Pirjanian, *Behavior coordination mechanisms - state-of-the-art*, Tech. Report IRIS-99-375, Institute of Robotics and Intelligent Systems, Univ. of Southern California, 1999.

[15] P. Pirjanian, T. Huntsberger, A. Trebi-Ollennu, H. Aghazarian, H. Das, S.S. Joshi, P.S. Schenker, "CAMPOUT: a control architecture for multirobot planetary outposts". In: *Proc. SPIE Conf. Sensor Fusion and Decentralized Control in Robotic Systems III*, USA, Nov. 2000.

[16] H. Schafer, M. Proetzsch, K. Berns, "Stereo-Vision-Based Obstacle Avoidance in Rough Outdoor Terrain", *International Symposium on Motor Control and Robotics*, 2005.

[17] C. Schlegel, J. Illmann, H. Jaberg, M. Schuster, and R. Worz, "*Integrating vision-based behaviors with an autonomous robot. Videre*", MIT-Press, ISSN 1089-2788, vol. 1, no. 6, Winter 2000.

[18] K.U. Scholl, J. Albiez, B. Gassmann. MCA - *An Expandable Modular Controller Architecture*. Forschungszentrum Informatik an der Universitaet Karlsruhe (FZI).