3D Real-Time Stereo Reconstruction for Small Humanoid Robots

Denis Klimentjew, Andre Stroh, Houxiang Zhang, Jianwei Zhang
Dept. of Informatics, University of Hamburg, Vogt-Kölln-Straße 30, 22527 Hamburg, Germany
{*klimentjew, 1stroh, hzhang, zhang*}*@informatik.uni-hamburg.de*

***Abstract*—This paper presents an application of 3D real-time stereo reconstruction for small humanoid robots and introduces the related issues like system integration and software architecture. First, the reconstruction and the calibration of two single cameras and a stereo camera system are presented. The accuracy of the control parameters is determined in the sub pixel range while reconstruction is based on several successive steps. After that, the depth information of the environment cannot only be visualized, but also used for localization and navigation as well as collision avoidance in a real-time way. The complexity of the application depends on the resolution of the camera or the selected region of interest (ROI). The application offers many different possibilities in real time for those humanoid robots which have no other sensors for the perception of their surroundings except cameras. Therefore the perception of the environment as well as robot-robot interaction can be realized. At the end of the paper we present our conclusions.**

***Index Terms*—3D reconstruction, real-time, stereo vision, software architecture.**

I. INTRODUCTION

The exploitation of depth information from stereo image pairs has been one of the most important topics in image processing for many years. The aim of this work was the design of an application for 3D reconstruction for small humanoid robots. The platform we used consists of two Hoap-2 robots. The robots bought from Fujitsu Automation Limited are used as a development platform at the TAMS group (Technical Aspects of Multimodal Systems) at the University of Hamburg. The robots are each equipped with a stereo camera system, an acceleration and gyro sensor as well as pressure sensors. Stereo camera systems are among the the most important methods to perceive the environment and are used by most humanoid robots, like QRIO [1] by Sony or Honda's ASIMO [2]. The Hoap-2 and the stereo camera system are presented in section III. However, the excellent perception ability is also the system's biggest disadvantage, as this large amount of available information needs to be captured and evaluated quickly. But the advantages are huge, as the perception of colours, facial detection and human-robot interaction are only possible with a camera. Comparable to the perception of humans, a stereo camera system offers a large amount of information, and the acquired image data can be used in many different ways. More than 80% of all human impressions of the environment are obtained by seeing. This is the most researched topic of computer vision.

First the related worldwide research will be examined. We will outline the most important work and compare the results.

A new method for 3D reconstruction from stereo images is presented in [3]. This approach resembles ours except that the resulting application uses a geometrical rectification of the images. The resulting distance estimates are better than those of normal 3D reconstruction. In our work, we compute rectificatio matrices using the procedure of Fusiello, Trucco and Verri [4]. We achieve unambiguously better results in the distance up to 2 m, a very important range for human-computer-interaction, perception and interaction with the environment. Unfortunately, the authors give no benchmarks for the processor time nor the construction of the resultant application, therefore their results cannot be compared with ours.

The 3D reconstruction in publication [5] is based on amounts of single steps, like the Canny edge detector, the sum of absolute differences (SAD) as well as the successive reconstructions of several levels. The results have a very high quality, but the calculation still takes a long time. For only three level reconstructions the application needs more than 21.6 seconds. The application is not a real-time program, moreover, very distant recumbent objects are much less interesting for a humanoid robot than objects in its immediate surroundings. Furthermore, it is very important that the robot should be able to react very fast to inside and outside events.

The authors of publication [6] introduce a real-time method for 3D reconstruction. However, the method requires special assembly, as it changes the balance of the humanoid robot. Moreover, the required mirrors narrow the reconstruction plain, so that for the reconstruction of the complete field of view several image pairs will be taken and reconstructed. Though the performance of the time is astonishing, the results are also not compared to the real distances.

Starting from a concrete problem, we searched for the quickest method which delivered the best results for the depth information. The aim was a real-time procedure which rendered navigation and the interaction with the environment possible. We tried to select the fastest possible method with the best algorithms for complete 3D reconstruction. Also, the calibration of the cameras played a very big role, because the data and their accuracy are usually very important for reconstruction. The result was an application which respectively delivers a 2.5D-Map or an array and assigns the third dimension and the distance to every pixel. We used OpenGL

to visualise the results. To accelerate the application, it was implemented in C++ by using some algorithms from the OpenCV library. OpenCV [7] is a free computer vision library, based on Integrated Performence Primitive (IPP) [8], optimised for Intel processors and co-developed by Intel. Before the application can be launched, first the cameras are calibrated individually, then the complete stereo camera system is calibrated. If the required parameters are known, we can according to our modular software architecture, introduced in section III. In the next step, the gained data can be used for localisation, navigation, etc. The resulting application has a steady complexity and is in real-time. The complexity depends only on the resolution of the images, and it also makes the programme suitable for collision avoidance.

The rest of this paper is structured as follows. In the next section we describe the implementation of camera calibration, rectification, 3D reconstruction as well as the visualisation. Then in section III the modular sofware architecture is introduced and discussed. The experimental results are introduced in section IV. In the last section, we conclude with the summary and discussion of the results as well as an outline of future work.

II. IMPLEMANTATION

For the implementation, the fastest algorithms with the best results were used. We decided on the C/C++ programming language and some OpenCV procedures. The resulting application consists of several steps. The first step, which is independent of the application, is the calibration of single cameras as well as the stereo camera system. After the camera parameters are determined, the main application can be launched. Primarily we rectified the captured images. In addition, we used rectification matrices, which are computed by camera calibration. Then we averaged the five images, to inhibit the noise, and afterwards smoothed them with a Gauss-Filter. The stereo image pairs were then searched for the corresponding points and attained a disparity map. We compared three different algorithms for the calculation of disparity. After this, the distance between the camera system and any object can be calculated. In the following, we will introduce and explain the single steps of the application.

A. CAMERA CALIBRATION

The calibration of stereo camera systems is a classical problem of stereo vision. There are several methods in the computer vision literature. The calibration requires an intrinsic, extrinsic and the lense destortion parameters of both cameras. For the 3D reconstruction, the distance between the cameras and their orientation in relation to each other are necessary factors. We implemented one of the possible rectification methods. The rectification not only yields the required data, but also immensely accelerates the later reconstruction.

The more accurate the determined intrinsic and extrinsic parameters, the more exact the computed distance. There exist several different procedures for camera calibration. These are distinguished on the basis of place, time or by using differen calibration bodies. For more details see [9].

At the end of an intensive search, there were only two procedures left to choose from under the given conditions, the procedures by R. Tsai [10] and Z. Zhang [11]. The procedure of Tsai requires at least seven corresponding point pairs which are not from the same level and are known before the calibration. This is a very big disadvantag, as the point pairs and their positions must be precisely determined before calibration . The procedure of Zhang uses a calibration body, which is recorded from different viewpoints. This causes the procedure to be very adaptable and to require only one calibration body with the known masses. In this work, we used the procedure by [11] for the calibration of the single cameras. We used a checkered calibration pattern with a single square size of 30×30 mm. The different distances between the camera and calibration body and the orientation of the calibration body as well as the number of the image pairs influence the accuracy of the calibration. The procedure is fully automatic and can be divided into three steps. In the first step, the homographie is estimated and finally the results are optimised with the method of Levenberg-Marquard [12]. The method is demonstrated in equation (1).

$$\sum_{j=1}^{n}\sum_{i=1}^{m} \|P_{w_ij} - P_c \cdot (A, k_1, k_2, R_j, T_j)\|^2 \qquad (1)$$

where the P_{w_ij} and P_c are a world coordinate point and its projection on the image level. For factors A, R and T the estimated values will be set. The coefficients k_1 and k_2 are set to zero at the beginning of the optimisation. Then the intrinsic and extrinsic parameters are determined from the homographie. With the modelling of the lense distortion and optimisation of the results the procedure is successfully concluded. The procedure is fast, stable and easy to use.

B. RECTIFICATION

We implemented the method by Fusiello, Trucco and Verri [4] in C/C++ for the rectification. The algorithm is compact and easily reproducible. Besides, the projection matrices of the left and right camera serve as input. Both projection matrices derive from intrinsic and extrinsic parameters of the respective camera and are computed after the calibration of the single cameras described in equation (2).

$$P = A \cdot [R\,T] \qquad (2)$$

where A is the matrix with the intrinsic parameters; R is a 3×3 rotation matrix computed with the help of the Rodrigues function and T is a 1×3 translation vector.

The optical centres of both cameras are preserved in this procedure, merely the rotation of the cameras is changed. The authors of [4] use this method to construct an ideal stereo camera system, while the projection plains of both cameras are rotated around the camera centres into the desired position. The approach is detailed in fig. 1. With this process it is possible to determine a new pair of projection matrices.

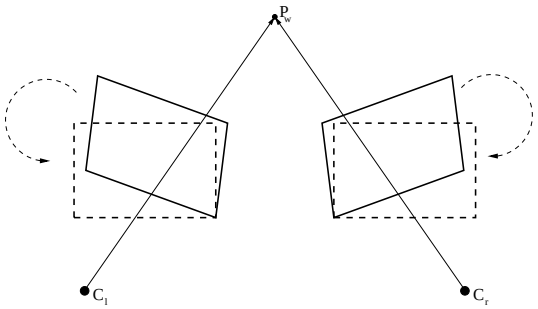

Fig. 1. Artificially contrived position of the image levels, which is reached by the rectification.

Then the transformation matrices, which project the provided matrices on the new image level, are calculated. These are applied to the images after lens distortion to correct them.

The resulting rectified images can be directly used for the reconstruction of the depth information. The process of the rectification allows for an efficient detection of corresponding pixels, because it reduces the search to one image line.

C. DEPTH RECONSTRUCTION AND MODELLING

In the literature two basic methods for 3D reconstruction exist, contour-based and surface-based reconstruction. With contour-based reconstruction, the typical characteristics must be found, for example edges or regions. The surface reconstruction delivers the depth information for the complete image.

Fig. 2. Pair of stereo images of a scene.

On the one hand, the contour-based methods are fast and will not affect the whole performence of our application. On the other hand, the complete information is often needed for human-robot interaction, as in the case of path planning. So we prefer the surface-based procedure, which has no without disadvantages for the performance of the complete application. To realise depth estimation in the best way possible, some well-known algorithms are compared. To test the advantages and disadvantages of the algorithms, their behavior is examined in the context of a workday scenario.

The input images of the reconstruction system are shown in the experimental setup of fig. 2. Objects have been equally positioned in different depths. To allow a fair comparison between different corresponding algorithms, these input images and their equalizated rectified image pairs (see fig. 3) are used henceforth.

Fig. 3. Equalizated and rectified pair of stereo images. This pair of images serve as input to successive correspondance analysis procedures.

At first, the Shirai algorithm is implemented and examined. A theoretical description of this algorithm can be found in [13]. The method is a primarly contour-based algorithm which was developed for detecting the correspondences of edge images. The algorithm searches corresponding point pairs with the help of the concentration of pixels within a window. Even if it leads to very good results for edges and texturised areas, it fails when used to reconstruct the depth for homogenous regions if it is used for surface-based reconstruction. The problems of this method become clear in fig. 4. Because all pixel pairs are compared, the complexity increases extremely with the resolution of the image.

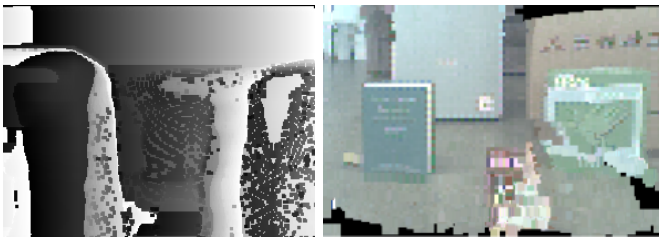

Fig. 4. Disparity map (left) based on the Schirai algorithm and on depth reconstruction (right) by using OpenGL.

Second, the Block-Matching algorithm is examined. As the first the authoritative pair of stereo images is divided into blocks of equal size whose block disparity is calculated. Then the block disparity on the pixel level is refined. The search for corresponding blocks is effected via different metrics. For our experiments, the Block-Matching algorithm has been implemented with the ”sum of the absolute differences” (SAD) metric.

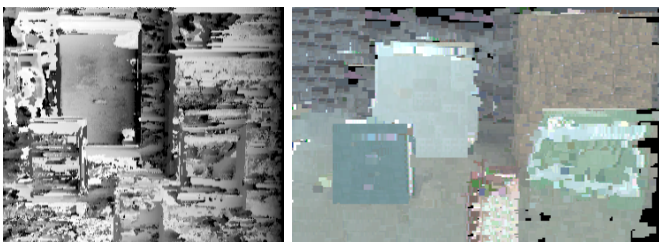

Fig. 5. Disparity map (left) based on Block-Matching. The brightness of an image point represents the 01depth of a real-world point. The brighter the point, the closer to the camera it is. A depth reconstruction (right) from the depth map is implemented by using OpenGL.

This algorithm shows a similar behavior as the Shirai algorithm through comparing intensity values of a certain window and the local searching character. The results and the complexity are also very similar to the Shirai algorithm concerning object edges, homogenous regions and hidden areas. In contrast to the Schirai algorithm, the edges of objects are rather blurred due to the refinement of the block disparity at pixel level. In areas with steadily recurring patterns, the grain of the algorithm fails completely, (see fig. 5).

Finally, the algorithm of Birchfield and Tomasi [14] is considered. The algorithm is double-stage, fast, based on dynamic programming and also part of the OpenCV library, so that this can be implemented immediately in our architecture.

Fig. 6. Disparity map (left) based on the dynamic programming algorithm by Birchfield & Tomasi [14]. The brightness of an image point represents the depth of the real-world point. The brighter, the closer to the camera. A depth reconstruction (right) from the depth map is implemented by using OpenGL.

In contrast to the above-mentioned correspondence procedures, the algorithm of Birchfield and Tomasi has a global searching character through its dynamic programming principle instead of local windowing functions. The results are shown in fig. 6. The algorithm scans a rectified imageline from a certain point, defined by the coordinate in the left image, for local optima and refines it under consideration of the depth values above and below the optima. This is realised via a compact cost function, described in equation 3.

$$\gamma(M) = N_{occ}k_{occ} - N_m k_r + \sum_{i=1}^{N_m} d(x_i, y_i) \qquad (3)$$

where M is a match sequence, k_{occ} is the constant occlusion penalty, k_r is the constant match reward, $d(x_i, y_i)$ is the dissimilarity between the x_i and y_i , and N_{occ} and N_m are the number of occlusions and matches, respectively, in M.

This cost function compensates steady disparities. Thus objects in the same level are better reconstructed than as curvatures or blurred crossings. The stereo algorithm is not as time-consuming and computationally expensive as classical methods that normally compare the similarity of frames of size $N \times N$ around pixels to determine their correspondence. Among the tested correspondence analysis procedures, the algorithm of Birchfield and Tomasi delivers superior results. The method is preferable to both of the other procedures and hence was included in our software architecture. Nevertheless, it is possible that under different test conditions, e.g. real natural scenes, the Schirai algorithm will lead to better results than Block-Matching or Birchfield. This could be due to the fact that most natural scenes only contain few continuous surfaces or repeating patterns.

After considering the disparity maps of the different algorithms, they were visualised three-dimensionally with OpenGL. The results of the reconstructed scene in 3D are shown in the right images beside the disparity maps respectively.

In the next section we introduce our software architecture. The architecture is a basis for our application and offers an interface for the other algorithms and methods.

III. MODULAR SOFTWARE ARCHITECTURE

The long-term objective of our project was to allow the Hoap-2 robot to interact with the environment and humans. The difficulty was due to the fact that the Hoap-2 has no other sensors for the perception of the environment than two cameras, as described in section I. The cameras are the Logitech Quickcams with a maximum resolution from 324×248 pixels. Both cameras are connected by USB 1.1 and a USB cable to one or several computers and are fastened on a 2-degree-of-freedom (DOF) pan-tilt unit with a base distance of approximately 61 mm. The robot, both cameras and the pan-tilt unit are presented in fig. 7.

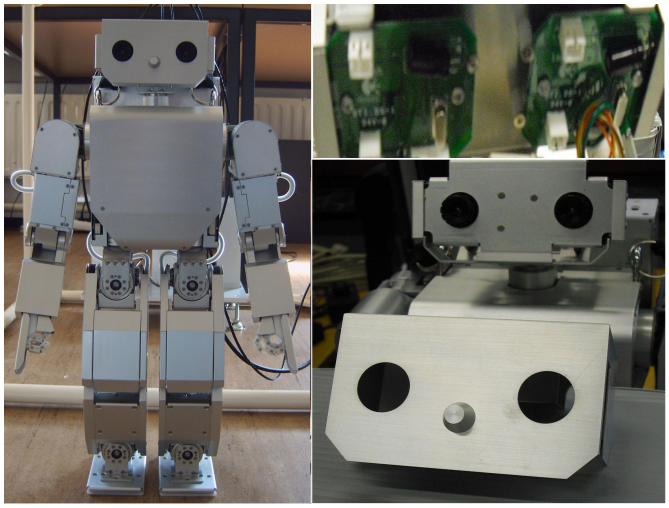

Fig. 7. Hoap-2 robot of the Fujitsu Automation Limited and the two USB Logitech Quickcams in the robot head with a 2-DOF pan-tilt unit.

The two cameras are united in a stereo camera system in this work. Some disadvantages become clear immediately, as the synchronisation of the cameras is not possible under the given conditions. In addition, the focus of the cameras can only be changed by hand. This disadvantage is tolerable as the advantage gained by the large amount of information is huge. Furthermore, as described in the previous section, the other manufacturers of robots also use stereo camera systems as the most important source of information for their humanoid robots.

It was our aim to design a software architecture which can also be used on different development platforms. This

should render the platform independent from the inserted stereo camera systems as well as the appointed algorithms and methods. The resulting modular software architecture is introduced in fig. 8.

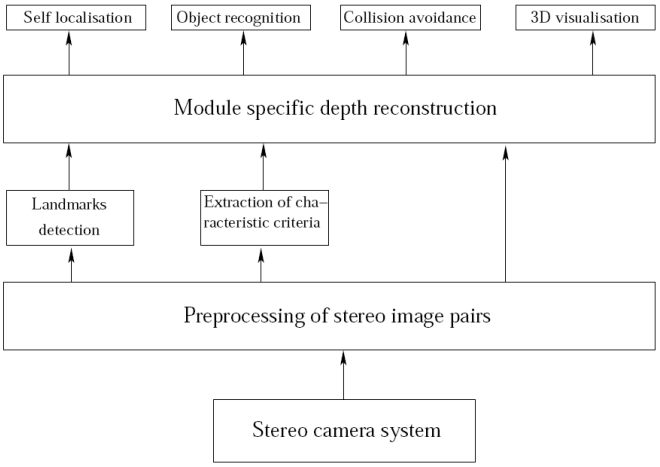

Fig. 8. The modular software architecture for the perception of and interaction with the environment.

The architecture needs no special hardware component, moreover, it is an application software and is programmed in C/C++. As a result, the application is universally deployable after compilation. The modular construction allows for the addition, removal or change of single applications without having to adapt the remaining components. The interface firmly determines the incoming and outgoing data. This makes the platform especially attractive for the testing of algorithms and procedures encased in single modules.

The application based on the architecture yields the depth reconstruction of the whole environment or a region of interest (ROI). With these data, it becomes possible to realise various objectives in robotics, like collision avoidance, self-localization, object recognition, 3D visualisation, *etc*.

The performance depends on the used components. If single modules have a steady complexity, then real-time will be guaranteed for the complete application.

IV. EXPERIMENTAL RESULTS

In this section we summarise the results of this work. We developed a real-time application based on our architecture, wich is introduced in section III. We examined the performance of the application with three different non-synchronised stereo camera systems[1], Logitech Quickcams with an image resolution of 324x248 pixels, two Sony DFW-VL 500 with 640x480 pixels, and two Sony XC-999P with a resolution of 704x576 pixels.

The temporal performance depends only on the resolution of the cameras or selected region, see fig. 9. The required

[1]All tests were run on a DELL Optiplex 745 with 1 GB ram, Intel Core 2 CPU @ 2.13 GHz, 2MB Cache, and ATI Radeon X1300 equipped with 256 MB internal memory.

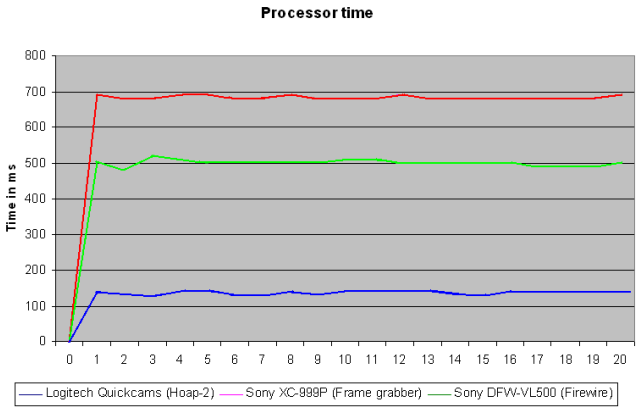

Fig. 9. Line chart of measured CPU-time for three different stereo camera systems for the following processing steps: lens distortion, rectification, image enhancement (Gaussian), disparity map incluing depth calculation.

CPU-time confirms our supposition that this is a real-time system. There are no outliers and the deviation from the average is very low.

TABLE I

Real distance	0,5 m	1,0 m	1,5 m	2,0 m	2,5 m	3,0 m
Calculated distance	0,5228	1,0257	1,5024	2,1364	2,5671	3,1364
Error in measurement	0,0228	0,0257	0,0024	0,1364	0,0671	0,1364

By reconstruction, the depth information can be computed. The approximated distances depend on the quality of the camera calibration as well as the calibration of the stereo camera system. The fully automatic calibration procedures we implemented deliver the parameters of single cameras as well as those of the stereo camera system, and also rectification matrices which accelerate the search for corresponding points enormously. The inserted parameters yield satisfactory results in spite of the smaller resolution of the Hoap-2 cameras and low wide angle, see table I and fig. 10.

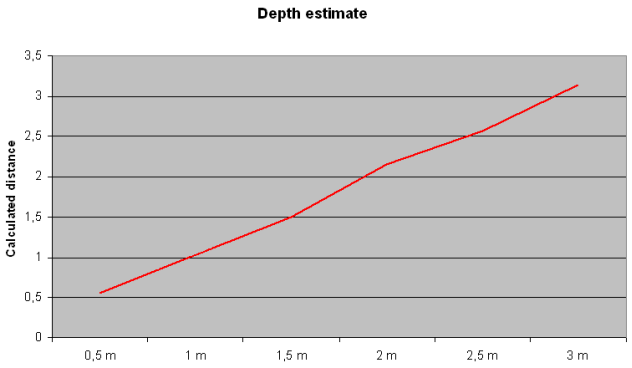

Fig. 10. Visualisation of the depth estimate with both Logitech Quickcams and by calibration to certain parameters.

The application we created can be used for all purposes of humanoid navigation or on other robots with stereo camera

systems.

To test the possibilities of our application, we implemented a simple collision avoidance on our Hoap-2. First, the obstacle is recognised. With the help of its position in the image, the Hoap-2 will decide whether to avoid the left or the right side. Because the robot strongly vibrates, the calculation takes place off-line. We calculate the perpendicular from the robot to the obstacle by using trigonometric functions. We thus compute the actual size and position of the object and recognise the impending collision. At that moment, the robot moves only statically under use of zero moment point (ZMP) [15], so that the maximal rotation angle per step is only 15 degrees. We use the maximal angle and calculate the step length for the avoidance, smaller steps entail a smaller movement radius and vice versa.

Fig. 11. Execution of the collision avoidance. The view of Hoap-2 by start and the resultant trajectory are displayed in the left lower corner of the image A and in the right lower corner of the image D.

With the computed data and reccurent neuronal network (RNN) [16] we generated a script, which is executed via the Fujitsu provided GUI. The execution of collision avoidance is shown in fig. 11. The robot moves from step A to D. The view of the Hoap-2 at the start is illustrated in the left lower corner of image A. The resultant trajectory is displayed in the right lower corner of image D.

As expected, the application ran robustly and delivered satisfactory results which are absolutely sufficient for other algorithms and methods of navigation for small humanoid robots.

V. Conclusion

This paper proposes a real-time application for depth reconstruction. In spite of non-synchronised stereo camera systems and lower resolution, the depth information is determined reliably. As a result, our application allows the realisation of all necessary parts of navigation. This application is particularly interesting for humanoid robots, because they often have only a stereo camera system for perception and the interaction with their environment.

Moreover, the architecture and the application based on it permit an ideal platform for tests with more different software components. Thus different components can be used, tested and changed without modifying the remaining structure.

References

[1] Sony. Qrio. *http://www.sony.net/Products/cx-news/vol32/sideview.html, last call 11/09*, 2008.

[2] Honda. Asimo. *http://world.honda.com/ASIMO/, last call 11/09*, 2008.

[3] A. J. Gallego, R. Molina, P. Compañ, and C. Villagrá. 3d reconstruction and mapping from stereo pairs with geometrical rectification. *Advances in Brain, Vision, and Artificial Intelligence, Springer Berlin / Heidelberg, ISBN: 978-3-540-75554-8*, 2007.

[4] A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *In: Mashine Vision and Applications, Springer Verlag*, 2000.

[5] H. Kim and K. Sohn. 3d reconstruction from stereo images for interactions between real and virtual objects. *Signal Processing: Image Communication 20, pp.61-75*, 2005.

[6] M. Vasiliu and F. Devos. Real-time 3d reconstruction on high resolution focal plane array. *In: Image Processing, 2000. Proceedings. 2000 International Conference on, pp.573-576*, 2000.

[7] Intel. Opencv library. *http://www.intel.com/technology/computing/-opencv/index.htm, last call 14/08*, 2008.

[8] Intel Software Network Intel Integrated Performance Primitives [IIPP]. *http://www.intel.com/software/products/ipp/ippvm20/index.htm, last call 14/08*, 2007.

[9] R. J. Valkenburg. *Classification of camera calibrations techniques*. 1995.

[10] R. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. *In: Proc International Conference on Computer Vision and Pattern Recognition, Miami Beach, Florida, USA 10*, 1986.

[11] Z. Zhang. A flexible new technique for camera calibration. *In: IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 2000.

[12] J. More. *The Levenberg-Marquard algorithm, Implementation and Theory*. Springer-Verlag, 1977.

[13] R. Klette, A. Koschan, and K. Schluens. *Computer Vision.* 1996.

[14] S. Birchfield and C. Thomasi. Depth discontinuities by pixel-to-pixel stereo. *In: Proceedings of the 1998 IEEE International Conference on Computer Vision, Bombay, India*, 1998.

[15] M. Vukobratovic and B. Borovac. Zero-moment point - thirty-five years of its life. *In: International Journal of Humanoid Robotics, 1(1), 157-173*, 2004.

[16] R. Zaier and F. Nagashima. Recurrent neural network language for robot learning. *Submitted to 20th conf. of Robotics Society of Japan*, 2002.