Title	An interactive 3D sensor system and its programming for target localizing in robotics applications
Author(s)	Heikkilä, Tapio; Ahola, Jari M.; Viljamaa, Esa; Järviluoma, Markku
Citation	Proceedings of the IASTED International Conference, Robotics (Robo 2010), November 24 - 26, 2010, Phuket, Thailand
Date	2010
Rights	Copyright © [2010] IASTED. This article may be downloaded for personal use only

Proceedings of the IASTED International Conference
Robotics (Robo 2010)
November 24 - 26, 2010 Phuket, Thailand

AN INTERACTIVE 3D SENSOR SYSTEM AND ITS PROGRAMMING FOR TARGET LOCALIZING IN ROBOTICS APPLICATIONS

Tapio Heikkilä, Jari M. Ahola, Esa Viljamaa and Markku Järviluoma

Technical Research Centre of Finland, PO Box 1100, FI 90571, Oulu, Finland
tapio.heikkila@vtt.fi

ABSTRACT
Interactive robotics introduces high flexibility to robotic task execution relying on human intelligence and understanding. We present an interactive sensor system for robotic applications with an approach for easy and flexible planning and programming. Flexible programming results to an arbitrary set of reference features for object localizing (fitting interactive measurements to geometric models) and for this we have also extended our earlier locating algorithm by dynamic formation of the Jacobian of the measurement model. Our CAD based approach gives a general and easy to use approach for programming and executing human-robot interactive tasks where human acts as a part of the environment observer. Experimental results are also reported.

KEY WORDS
Interactive Robotics, Human Robot Interaction, Sensor programming

1. Introduction

Interactive Robotics is a control scheme where the human intelligence and human capability to observe unstructured or highly dynamic environments is utilized to enable modelling and execution of robotic tasks [1]. In interactive robotics human role varies from supporting measurements and observations to manual motion guidance by target or tool guidance [2]. Typically interactive robotics has its potential in very flexible production [3], where conventional full automation is not feasible. Improvement in working conditions and ergonomics is another major reason for applying interactive - or human-robot cooperative - robotics [4]. Manual handling or assembly of heavy parts covers a wide range of activities, including lifting, pushing, pulling and holding. Working operations including these types of work include often a substantial risk for injuries, especially musculoskeletal disorders [5], and then applying interactive robot technologies is a potential way to go. When the operators are released from carrying heavy loads, they can focus on guidance and controlling the tasks.

Our goal has been to utilize human intelligence and skills to reduce hazardous manual work and introduce flexibility to robotic tasks. We apply flexible measurement technologies with optical sensors, supervised by the human operator to adapt to variation in target object locations and further, for robot path adaptations. This is supported by CAD based programming of sensory operations. Interactive robotics, in the form of sparse environmental modelling has been introduced earlier, e.g., in [1]. However, industrial exploitation implies capabilities to handle complicated shapes and quick and easy to use commanding of the system, implying detailed reference information for the robot. The novelty in our work is especially in the easy programming of the sensor tasks, using standard CAD tools, and also the full application of the interactive sensor scheme with dynamic formation of the measurement model in a real experimental test environment. In the following chapters we give further details about the operating principles of the interactive sensor system, a programming scheme based on CAD tools, and results from practical tests in an experimental robot cell.

2. Operating principles of the interactive sensor system

The interactive sensor system is used for locating target objects for further handling or processing by robots. This is needed in many robotic applications, where the target object geometry is well known, but environmental conditions or the size or complicated shapes of the target objects does not allow the use of fully automatic sensor systems. The interactive sensor system consists of two cameras, and a laser pointer device, and as such follows the principles of existing industrial 3D measurement systems. When the human operator points 3D features with the laser pointer, e.g., surfaces of the target objects the camera system computes and records corresponding 3D points. When these are matched to a reference geometric model of the target object, the location of target

DOI: 10.2316/P.2010.703-056

object in the robot working space will be derived and used for on-line path updates.

The usage of the interactive sensor system implies proper planning and programming tools while preparing the tasks off-line and also support to easily verify measurements on-line. The target features need to be determined and listed as guidance and reference for the human operator. Also the measurement results wrt. these target features need to be illustrated on-line for the human operator for verification before the robot starts it operation.

The following use cases give an overall view of the required activities of using the interactive sensor system:

- Sensor to robot calibration: calibrate the sensor system (cameras) to the robot coordinate system
- Program the measurement tasks: prepare the robot station for executing appropriate measurement actions to locate a target object
- Locate the target: carry out the location measurements for the target object
- Execute the robot task: carry out the robotized processing task with working paths updated (e.g., update user coordinate frames within paths) by the measurement results

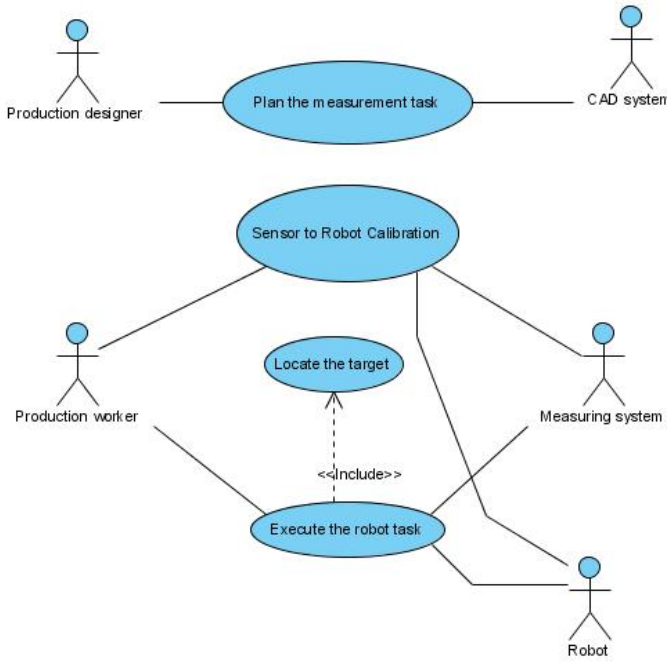

Fig. 1. Use cases of the interactive sensor system.

The planning of the measurement task by using a CAD – model is carried out with the following steps:

1. Human planner selects the reference features (eg. plane, cylinder, sphere)
2. The programming system determines the parameters of the reference features and extends the measurement model correspondingly.
3. Human planner sets the initial target object pose.

Localizing in the robot task execution is based on initial estimate of the target object location, set of reference features and the results from the interactive measurements. Detailed steps in localizing an object goes in the following steps:

1. Human operator and the measurement system interactively measure the 3D -points from the surfaces of the target object as defined by the reference features. To ensure feature-to-measurement correspondence, this measurement procedure is strictly guided by the measurement system: it shows to the operator graphically each feature from which to make the measurements one by one.
2. Localizing algorithm calculates, based on the measurement model, errors e_i and partial derivatives $\partial e_i/\partial \Delta$ for all measured points and gathers them in the error vector E and Jacobian matrix J
3. Localizing algorithm calculates the pose correction and updates the target object pose
4. If the pose correction is close to zero the localizing algorithm stops, otherwise it returns to step 2

3. Feature definition and target object locating algorithms

Selection of the reference features

The reference features are common geometric shapes such as planes, spheres and cylinders. The selected reference features should totally define the pose (location and orientation) of the target object. The pose of the target object is fully defined when all six degrees of freedom are defined. The basic geometric shapes alone are insufficient to define the target object pose because by nature a plane permits planar translation and rotation, a sphere allows 3D -rotations and a cylinder allows axial rotation and translation. A fully defined pose can be attained with a proper combination of these geometric shapes. For example, two coinciding planes allow one-dimensional translation but three coinciding planes define all degrees of freedom.

Parametrization of the target object

In the parametrization phase the selected reference features are identified from the CAD -model of the target object. A plane is defined with a point p_p and a surface normal vector n_p. A sphere is defined with a center point p_s and radius r_s. The parameters of a cylinder consists of a radius r_c and the axis of a cylinder which is defined with a point p_c and a unit vector s_c.

Measurement model

The measurement system consists of two cameras calibrated with respect to world coordinate system.

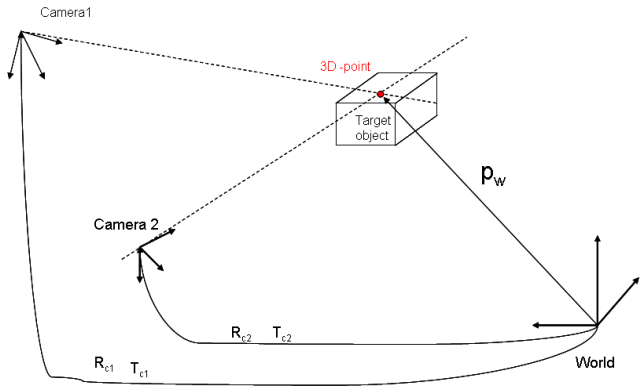

Fig.2. The principle for 3D measurements based on two cameras.

The target object is located in the both camera's field of view and all the selected reference features can be seen in both camera images. A 3D -point on the target object surface is measured in the image planes of the cameras. The camera specific 3D view lines are defined separately with a pinhole camera model in the local camera coordinate systems (Fig 3).

A 3D -line in the camera coordinate system is defined as

$$\overline{s_{ci}} = \overline{p_i} + k\overline{v_i} \quad (1)$$

in which $\overline{p_i}$ is a point in the image plane

$$\overline{p_i} = \begin{bmatrix} x_i & y_i & 0 \end{bmatrix}^T \quad (2)$$

The unit vector $\overline{v_i}$ defining the direction of the line is calculated as

$$\overline{v_i} = \frac{\overline{p_i} - \overline{p_f}}{\left|\overline{p_i} - \overline{p_f}\right|} \quad (3)$$

in which $\overline{p_f}$ is a camera specific point of focus

$$\overline{p_f} = \begin{bmatrix} 0 & 0 & f \end{bmatrix}^T \quad (4)$$

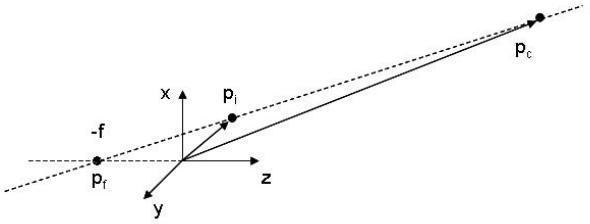

Fig 3. A pinhole camera model.

The 3D -lines are further transformed to world coordinate system

$$\overline{s_{wi}} = (\overline{R_c} \cdot \overline{p_i} + \overline{T_c}) + k \cdot \overline{R_c} \cdot \overline{v_i} \quad (5)$$

in which $\overline{R_c}$ and $\overline{T_c}$ define the orientation and location of the camera in the world coordinate system. The measured 3D -point is in the middle of the shortest line segment between the lines evaluated from separate cameras.

The localization principle

The aim of the localization task is to determine the target object pose that fits the measured 3D -points on the corresponding reference surfaces. We follow the algorithm originally given in [6] and extend it here by the dynamic formation of the Jacobian of the measurement model.

The pose of the target object is defined by a rotation matrix R and a translational vector T. The initial target object pose is updated to minimize the distances between measured 3D -points and reference surfaces (Fig 4.). The localization algorithm is iterative by nature and calculation is continued as the pose correction is close enough to zero.

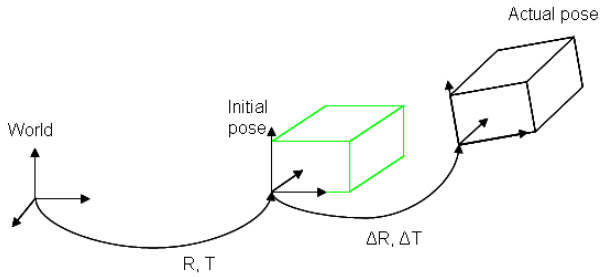

Fig. 4. The principle for localization algorithm.

The rotational correction $\overline{\Delta R}$ is defined by components $\Delta\varphi_x$, $\Delta\varphi_y$, and $\Delta\varphi_z$:

$$\overline{\Delta R} = \mathrm{Rot}(x, \Delta\varphi_x)\mathrm{Rot}(y, \Delta\varphi_y)\mathrm{Rot}(z, \Delta\varphi_z) \quad (6)$$

$$Rot(x, \Delta\varphi_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\Delta\varphi_x) & -\sin(\Delta\varphi_x) \\ 0 & \sin(\Delta\varphi_x) & \cos(\Delta\varphi_x) \end{bmatrix} \quad (7)$$

$$Rot(y, \Delta\varphi_y) = \begin{bmatrix} \cos(\Delta\varphi_y) & 0 & \sin(\Delta\varphi_y) \\ 0 & 1 & 0 \\ -\sin(\Delta\varphi_y) & 0 & \cos(\Delta\varphi_y) \end{bmatrix} \quad (8)$$

$$Rot(z,\Delta\varphi_y)=\begin{bmatrix}\cos(\Delta\varphi_z) & -\sin(\Delta\varphi_z) & 0\\ \sin(\Delta\varphi_z) & \cos(\Delta\varphi_z) & 0\\ 0 & 0 & 1\end{bmatrix} \quad (9)$$

The translational correction $\overline{\Delta T}$ is defined by three translational components

$$\overline{\Delta T}=\begin{bmatrix}\Delta t_x\\ \Delta t_y\\ \Delta t_z\end{bmatrix} \quad (10)$$

The correction vector $\overline{\mathbf{\Delta}}$ is defined as

$$\overline{\mathbf{\Delta}}=\begin{bmatrix}\Delta\varphi_x\\ \Delta\varphi_y\\ \Delta\varphi_z\\ \Delta t_x\\ \Delta t_y\\ \Delta t_z\end{bmatrix} \quad (11)$$

The correction vector is calculated

$$\overline{\Delta}=-(\overline{J}^T\overline{J})^{-1}\overline{J}^T\cdot\overline{E} \quad (12)$$

in which the vector $\overline{E}$ includes the location errors e_i of the measured points

$$\overline{E}=\begin{bmatrix}e_1\\ \vdots\\ e_i\end{bmatrix} \quad (13)$$

and $\overline{J}$ is the Jacobian matrix gathered from partial derivatives of e_i respect to $\overline{\mathbf{\Delta}}$

$$\overline{J}=\begin{bmatrix}\frac{\partial e_1}{\partial\overline{\Delta}}\\ \vdots\\ \frac{\partial e_N}{\partial\overline{\Delta}}\end{bmatrix} \quad (14)$$

in which N is the number of measured points.

The updated location T and orientation R of the target object pose are calculated as

$$\overline{T}=\overline{R}\overline{\Delta T}+\overline{T} \quad (15)$$

$$\overline{R}=\overline{R}\overline{\Delta R} \quad (16)$$

Formation of the Jacobian matrix

The reference feature geometries are defined in the target object coordinate system. Therefore the measured 3D -points are transformed to the target object coordinate system. A point in the target object coordinate system is

$$\overline{p_M}=\overline{R}^T(\overline{p_w}-\overline{T}) \quad (17)$$

in which $\overline{p_w}$ is the point in world coordinate system.

For all measured 3D -points a partial derivative representing a row of the Jacobian matrix is calculated as

$$\frac{\partial e_i}{\partial\overline{\mathbf{\Delta}}}=\frac{\partial e_i}{\partial\overline{p_{Mi}}}\frac{\partial\overline{p_{Mi}}}{\partial\overline{\mathbf{\Delta}}} \quad (18)$$

in which e_i denotes localization error calculated for reference features respectively.

If the reference feature is a plane the localization error e_i is the distance to the plane surface

$$e_i=\frac{(\overline{p_{Mi}}-\overline{p_p})^T\overline{n_p}}{\sqrt{\overline{n_p}^T\overline{n_p}}} \quad (19)$$

and the partial derivative of e_i respect to p_{Mi} is

$$\frac{\partial e_i}{\partial\overline{p_{Mi}}}=\frac{1}{\sqrt{\overline{n_p}^T\overline{n_p}}}\overline{n_p}^T \quad (20)$$

If the reference geometry is a sphere the error e_i is the distance to the surface of the sphere as follows

$$e_i=\sqrt{(\overline{p_{Mi}}-\overline{p_s})^T(\overline{p_{Mi}}-\overline{p_s})}-r_s \quad (21)$$

and the partial derivative of e_i respect to p_{Mi} is

$$\frac{\partial e_i}{\partial\overline{p_{Mi}}}=\frac{1}{\sqrt{(\overline{p_{Mi}}-\overline{p_s})^T(\overline{p_{Mi}}-\overline{p_s})}}(\overline{p_{Mi}}-\overline{p_s})^T \quad (22)$$

If the reference geometry is a cylinder the error e_i is the distance to the surface of the cylinder:

$$e_i=\sqrt{(\overline{p_{Mi}}-\overline{p_c})^T\overline{\mathbf{A}}^T\overline{\mathbf{A}}(\overline{p_{Mi}}-\overline{p_C})}-r_c \quad (23)$$

in which $\overline{\mathbf{A}}$ is a 3x3 square matrix calculated as

$$\overline{\mathbf{A}} = \frac{\overline{\mathbf{s}_S}\,\overline{\mathbf{s}_s}^T}{\overline{\mathbf{s}_s}^T\,\overline{\mathbf{s}_S}} - \mathbf{I}_{3x3} \tag{24}$$

and I_{3x3} is a 3x3 identity matrix. The partial derivative of e_i respect to p_{Mi} is

$$\frac{\partial e_i}{\partial \overline{p_{Mi}}} = \frac{1}{\sqrt{\left(\overline{p_{Mi}} - \overline{p_c}\right)^T \overline{\mathbf{A}}^T \overline{\mathbf{A}} \left(\overline{p_{Mi}} - \overline{pc}\right)}} \left(\overline{p_{Mi}} - \overline{p_c}\right)^T \overline{\mathbf{A}}^T \overline{\mathbf{A}} \tag{25}$$

.The partial derivative $\frac{\partial \mathbf{p}_{Mi}}{\partial \mathbf{\Delta}}$ is defined as

$$\frac{\partial \mathbf{p}_{Mi}}{\partial \mathbf{\Delta}} = \begin{bmatrix} \frac{\partial \mathbf{p}_{Mi}}{\partial \Delta\varphi_x} & \frac{\partial \mathbf{p}_{Mi}}{\partial \Delta\varphi_y} & \frac{\partial \mathbf{p}_{Mi}}{\partial \Delta\varphi_z} & \frac{\partial \mathbf{p}_{Mi}}{\partial \Delta t_x} & \frac{\partial \mathbf{p}_{Mi}}{\partial \Delta t_y} & \frac{\partial \mathbf{p}_{Mi}}{\partial \Delta t_z} \end{bmatrix} \tag{26}$$

$$\frac{\partial \overline{p_{Mi}}}{\partial \overline{\mathbf{\Delta}}} = \begin{bmatrix} 0 & -z_{Mi} & y_{Mi} & -1 & 0 & 0 \\ z_{Mi} & 0 & -x_{Mi} & 0 & -1 & 0 \\ -y_{Mi} & x_{Mi} & 0 & 0 & 0 & -1 \end{bmatrix} \tag{27}$$

in which x_{Mi}, y_{Mi} and z_{Mi} are the coordinates of p_{Mi} in the target object coordinate system.

4. Programming the interactive measurement tasks

Reference surface features are programmed using CAD software. They are acquired from the CAD system simply by picking points from a suitable target surfaces. Currently it is the user's responsibility to select the features in such a way, that the quality of the locating operation is acceptable. Rules of thumb, like "cover all 3 directions with the surface normals of the reference features" are used. While picking the points, the user names the related reference feature, after which the geometric reference parameters of the feature are calculated in the object coordinates by fitting the points in the corresponding shape model and a corresponding row is added into the Jacobian of the measurement model.

Figures 5 and 6 illustrate picking and showing the reference features from the 3D object model using the pick point tool of MeshLab 3D –viewing software. Minimum amount of picked points needed to identify a reference feature depends on the geometry of the feature. For example, a plane needs at a minimum 3 points, sphere 4 points and cylinder 5 points.

Minimum amount of measured points needed to locate the object depends on the amount and quality of the reference features. Minimum requirement is usually 3 points per surface to tolerate a measurement noise.

The measurement results should also be illustrated for the human operator so that the results can be verified before the system is trigged to start the execution of the handling or processing operations. Because the shapes of the target object can be complicated, the knowledge of the human planner/programmer is used to select a sparse set of verification features from the CAD model, which can be shown to the human operator of the robot. These verification features are simply 3D points in the form of a list, which are superimposed as a set of 3D line segments into the measured images, visualizing the quality of the measurements and model fit to the measurements.

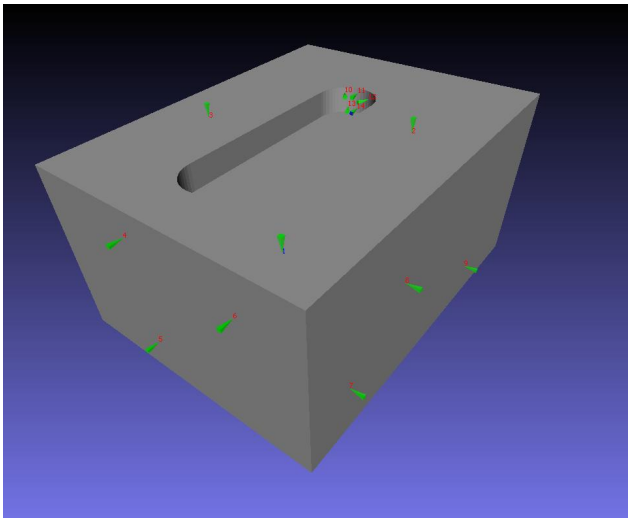

Fig 5. Picked reference points for determining the parameters of the target features.

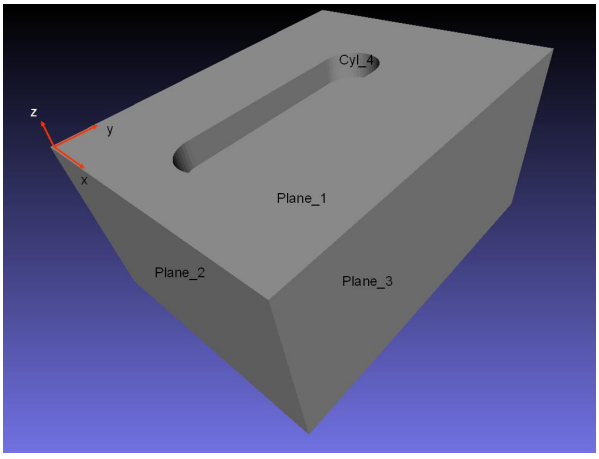

Fig 6. Reference shapes for the example object.

5. Experimental tests

A prototype interactive sensor system has been constructed and its functionality has been tested in practice. The system architecture is illustrated in Fig 7.

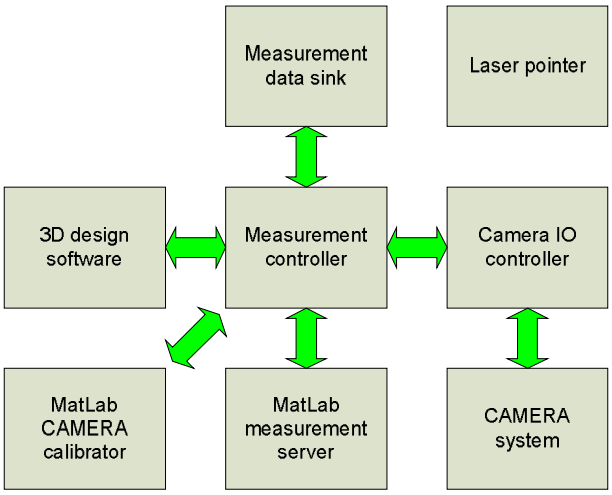

Fig.7. System architecture of the experimental interactive sensor system..

Component descriptions

The main test system components are characterized as follows.

Measurement controller
- Visual C++ graphical user interface on Windows XP
- Reads measurement commands from the wireless pointer
- Reads images through camera I/O controller
- Sends images to the MatLab measurement server through Mex-functions
- Receives object's updated pose data and sends it to the sink

Camera I/O controller
- Basler Pylon camera controller libraries on C++ which are embedded to the measurement controller
- Acquires and codes images using Bayer filter

Multi-camera system
- 2 Basler SCA 1600 cameras with Ethernet connections
- 1600 x 1200 pixels
- Tamron 16 mm objectives

3D design software
- MeshLab 3D-modelling software
- picks surface points from the 3D –model from which to calculate the reference features

Laser pointer
- off-the-shelf wireless Logitech laser presentation pointer
- "next page" button interruption flag is read in measurement controller as image acquisition trigger signal

MatLab measurement server
- reads RBG –images and uses camera internal calibration data to do pixel correction
- detects red channel pixel clusters between predefined threshold values based on red channel intensity and calculates the center point of the laser spot
- calculates world coordinate value based on corrected pixel coordinates
- fits world coordinate values to the reference model and calculates new pose to the object
- sends pose or coordinate transformation to the measurement sink

Measurement sink
- an entity exploiting the measured pose or transformation for further use
- could be i.e. robot controller etc.

Calibration for the internal camera parameters

The camera system internal parameters were calibrated using an open source MatLab toolbox [7]. The toolbox used a chessboard as a reference piece. Twenty images of the chessboard were taken from different distances and angles by both of the cameras. The chessboard corners were later extracted using the interactive user interface. The corner pixel coordinates, chessboard square dimensions and amount with corner finder window size provided adequate initial data for calibration algorithm to automatically find every square corner pixel coordinates and calculate certain objective parameters like focal length, skew, both radial and tangential distortion and pixel error. Best calibration results were got using calibration tool iteratively, decreasing the size of the corner finder window gradually. The calibration data was directly used for pixel correction later in this study. The toolbox uses slightly modified estimation methods presented by Zhang [8] and intrinsic model described by Heikkilä and Silven [9].

Calibration of the both cameras to the chessboard coordinate system for stereo vision

The cameras of the measurement system were calibrated to form a stereo vision system in order to detect 3D points from the target object. Chessboard square corners acting as reference calibration points were extracted from the chessboard images from both cameras using the MatLab toolbox [7]. These pixel coordinates were corrected from the skew and distortions using the parameters from the internal calibration phase. The calibration method calculates two camera postures in chessboard coordinate system. Calculated camera postures were used later on converting two image points to one world coordinate point.

Acquiring 3D points

Reference features were indicated by the human operator to the sensor system using industrial cameras and off-the-shelf laser presentation pointer. Cameras were triggered by the human operator pressing the button of the standard laser presentation pointer connected wirelessly to the C++ graphical user interface. Images were pre-processed using the GUI and Basler camera libraries. Pre-processed images were sent to Matlab server for further processing using the Matlab mex interconnection functions. After the pixel correction, the red channel of the RGB image was observed for the laser spot detection. Using the specified threshold values for area size, certain red pixel cluster area was detected to be the laser beam spot and its center point was calculated, converted to the world coordinates and returned back to the GUI software as it can be seen in Fig 8. Depending on the geometry of the reference features, the image acquisition and processing phase was repeated as many times as necessary. Finally the collection of the measured points in certain order was sent to the pose estimation (or localizing) algorithm.

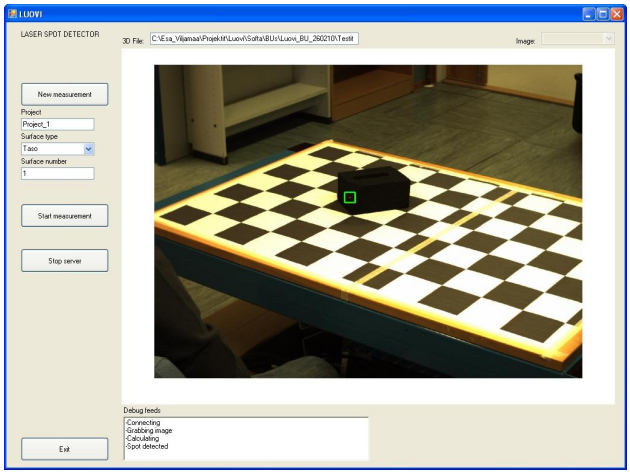

Fig. 8. GUI screen with image of the object and detected red laser spot inside the green rectangle.

Fitting measured 3D points to the reference features

The measured 3D points were matched to the corresponding reference features and the pose of the object in the robot base coordinate was estimated in the Matlab server. Verification features, as programmed from the CAD model are also shown as 3D line segments in the measured image. This is illustrated in fig 9.

The locating accuracy of the current sensor system depends on the quality of the reference features, but also on the camera resolutions and the imaging geometry. With the current settings and a feasible set of reference features (all degrees of freedom well covered) accuracy in the level of +/- 1 - 3 mm can be achieved. Our preliminary tests also support this estimate. In the tests, a block was located with interactive measurements, the block coordinate frame (user coordinate system) was updated in the robot program, and the top face was marked by the robot (ABB IRB 1400) with a green circle. Then the location of the test block was changed, its location was measured again, coordinates in the robot program updated and the marking was done again but with a black circle. The test was repeated several times. The circles coincide well, within 1 mm bounds; an example from one test run is shown in fig 10.

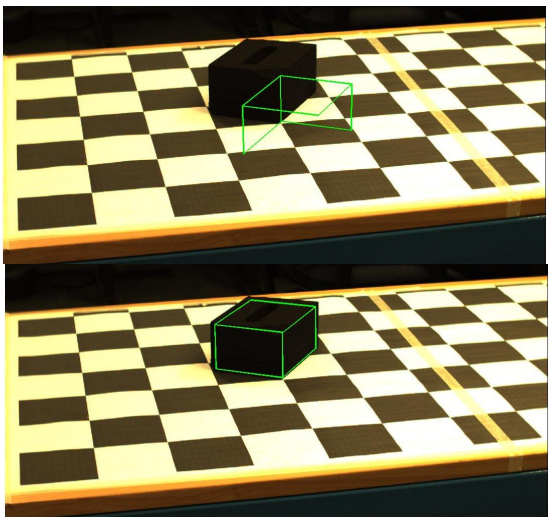

Fig. 9. Object locating result, illustrated with the verification features (green wireframe model).

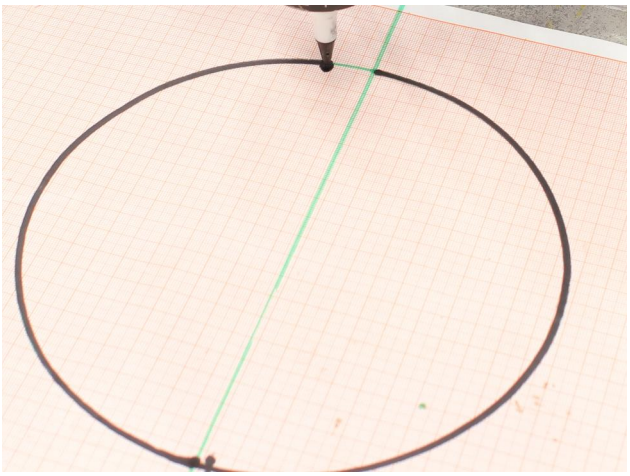

Fig. 10. Tests with robot operations: top face of an object was marked twice (green and black circles), based on two measurements for the target object location..

6. Discussion

The interactive sensor system has been implemented and is working and is easy to use. It is based mostly on commercial HW and SW tools, with extension to estimation of planned feature parameters and also estimation of target object location during task execution. The planning and programming of the sensor operations are done relying on general CAD tools, which is easy to

implement and use, and also applicable to sensor programming for automatic task execution. Our earlier locating algorithms were also extended correspondingly by dynamic formation of the Jacobian of the measurement model.

The Jacobian may have singular points when cylindrical and spherical reference features are used, if the measured point coincides with the center point of the sphere or the center line of the cylinder. This may happen during the iteration, and then the nominal values of the estimated parameters should be deviated from the singular case. In addition, if the noise level of the measured 3D point is in the same magnitude as the radius of the sphere or the cylinder the situation is the same, although practically this is not so realistic: e.g., in our case, a spherical reference feature with a 1 – 2 mm radius is not sensible.

Interactive task execution takes additional time, and extends the task execution time of the robot system in a range of several seconds to several tens of seconds. In many cases, especially in very flexible and close to one-of-a-kind production this can be really acceptable. On the other hand, due to environment complexity and frequent changes, this may be the only feasible way to introduce robotic technologies to heavy duty industrial applications, in many cases removing otherwise inevitable ergonomically problematic and hazardous working conditions.

7. Conclusions

We have presented an interactive sensor system for robotic applications with an approach for easy and flexible planning and programming. Our CAD based approach gives a general and easy to use approach for carrying out human-robot interactive tasks.

Acknowledgements

The work has been financially supported by TEKES – the Finnish Funding Agency for Technology and the Technical Research Centre of Finland (VTT), which is greatly acknowledged by the authors.

References

[1] Halme A., Heikkilä T., Torvikoski T., An Interactive Robot Control System. International Journal of Robotics and Automation, vol 2, No. 3, 1987, pp. 155 - 162.
[2] Koskinen J., Heikkilä T., Pulkkinen T., A monitoring concept for cooperative assembly tasks, in SpringerVerlag of Lecture Notes in Automation, Collaboration and EServices; the Edition of the Selected Papers entitled "Frontiers of Assembly and Manufacturing" from the 2009 IEEE International Symposium on Assembly and Manufacturing (ISAM 2009).
[3] R. Bernhardt, D. Surdilovic, V. Katschinski, G. Schreck, and K. Schröer, "Next Generation of Flexible Assembly Systems," in Innovation in Manufacturing Networks, Boston: Springer, 2008, pp. 279-288.
[4] P. Akella et al., "Cobots for the Automobile Assembly Line," in 1999 IEEE International Conference on Robotics & Automation, 1999, pp. 728–733.
[5] Code of Practice for Manual Handling (Occupational Health and Safety Act 1985). No. 25, 20 April 2000, Victorian Work Cover Authority, Australia. 68 p.
[6] Sallinen, M., Heikkilä, T., Estimation of the surface model parameters and analysis of spatial uncertainties. Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems. MFI2001. Baden-Baden, Germany, 20 - 22 Aug. 2001. IEEE. Germany (2001), pp. 209 – 214
[7] Bouguet, Camera calibratioin toolbox, http://www.vision.caltech.edu/bouguetj/calib_doc/
[8] Zhang; Z., Flexible camera calibration by viewing a plane from unknown orientations. The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999.Volume: 1. Pp: 666 – 673.
[9] Heikkila, J.; Silven, O.; A four-step camera calibration procedure with implicit image correction. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1997. Pp. 1106 - 1112