Proceedings of the OAGM&ARW Joint Workshop 2017 DOI: 10.3217/978-3-85125-524-9-14

A Visual Servoing Approach for a Six Degrees-of-Freedom Industrial Robot by RGB-D Sensing

Thomas Varhegyi, Martin Melik-Merkumians, Michael Steinegger, Georg Halmetschlager-Funek, and Georg Schitter

Abstract— **A visual servoing approach is presented that uses depth images for robot-pose estimation utilizing a marker-less solution. By matching a predefined robot model to a captured depth image for each robot link, utilizing an appropriate approximation method like the Iterative Closest Point (ICP) algorithm, the robot's joint pose can be estimated. The a-priori knowledge of the robot configuration, alignment, and its environment enables a joint pose manipulation by a visual servoed system with potential to collision detection and avoidance. By the use of two RGB-D cameras a more accurate matching of the robot's links is feasible while avoiding occlusions. The modeled links are coupled as a kinematic chain by the Denavit-Hartenberg convention, and are prevented from divergence during the matching phase by the implementation of an algorithm for joint pose dependency. The required joint orientation of the robot is calculated by the ICP algorithm to perform a pose correction until its point cloud align with the model again. First tests with two structured light cameras indicated that the recognition of the robot's joint positions brings good results but currently only for slow motion tasks.**

I. INTRODUCTION

The fourth industrial revolution involves the use of new robotic technologies for smart and efficient work-flows in an innovative way. Humans will work together with robots side-by-side and integrate them in their every day work life as a collaborative device. Therefore, a collision detection with humans and the environment has to be established, for instance, with pressure sensitive skins [1, 2] or abnormal force recognition [3, 4] which are two approaches for a collaborative aspect. Another idea is the integration of visual perception [5, 6]. Robots should see where they are, know and see the environment they move in and know how they can grab and move without disturbing the work-flow. The focus of this paper lies on the application of computer/-machine vision methods for image processing and robot actuation. Vision-based motion control of robots is called visual servoing, where the robot manipulator is operated by the evaluation of visual information from an eye-to-hand (camera fix to workspace position) or an eye-in-hand (camera attached to robot) composition [7]. Figure 1 shows the recording of a robot in an eye-to-hand composition, that is used for the visual servoing approach in this paper. The advantage of visual servoing is that the teach-in procedure of a robot can be omitted since tool-tip-pose errors caused by low accuracy between the tool-tip-pose and the joint angle

All authors are with the Automation and Control Institute (ACIN), Vienna University of Technology, A-1040 Vienna, Austria. Contact: melik-merkumians@acin.tuwien.ac.at (corresponding author)

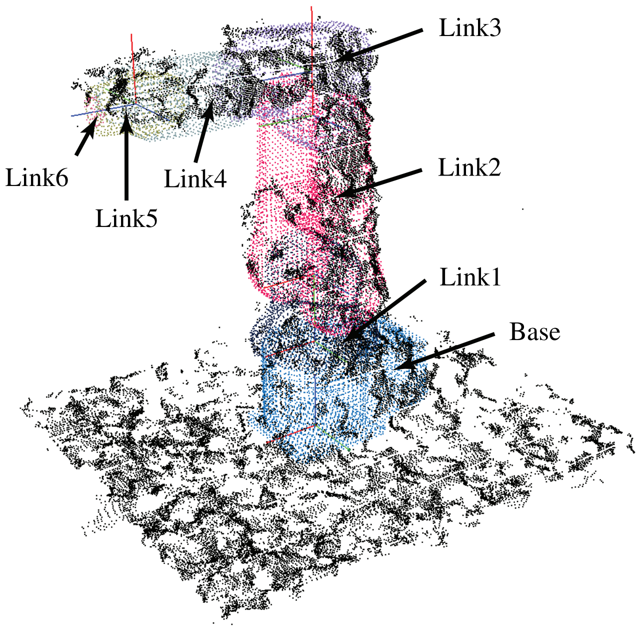
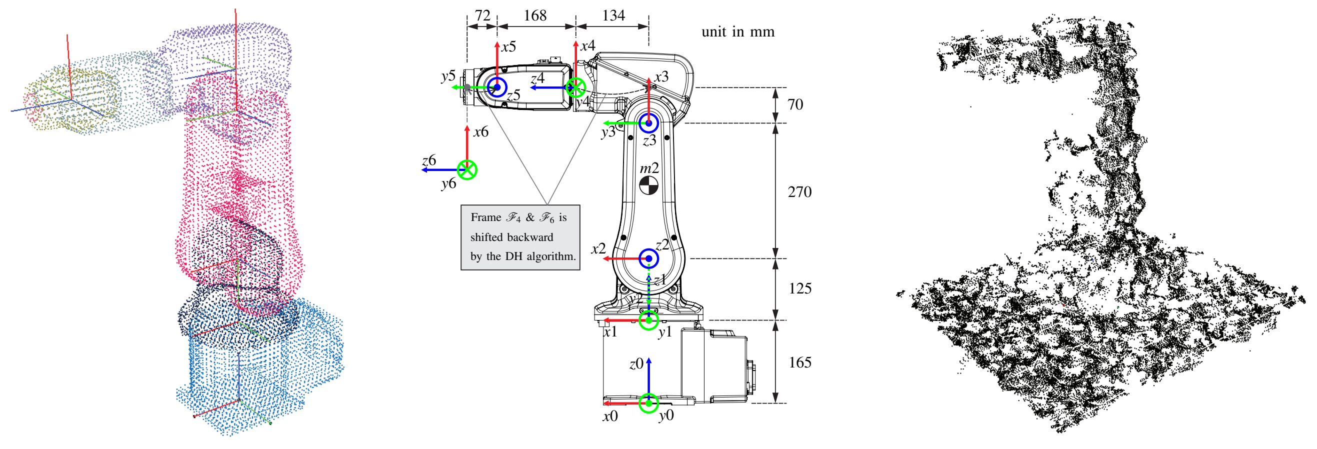


Fig. 1: ABB IRB 120 point cloud model overlaid by the captured point cloud from the Intel® RealSense R200.

can be corrected in addition. These visual information can be exploited as position- or image-based information [8–10]. Position-based detection uses interest-points in the image to detect the object position, while image-based detection uses a template image of the designed object to predict how the camera should be aligned to the object.

So far, mainly 2D cameras have been applied for visual servoing applications [11–13]. The accuracy of the interest point estimation in the image as edges or corners determines how precisely the robot can be positioned by 2D cameras. For objects without distinctive characteristics as curved shapes without edges, these kinds of camera systems do not suite perfectly. In this case depth sensing cameras is the better choice.

RGB-D imaging systems can be separated into three main groups. First, stereo vision systems [14] which are based on two cameras and feature disparity where the depth information is obtained by the use of triangulation. Second, structured light cameras [15, 16] with the same basic principles as stereo vision cameras but instead of the second camera a projector is used. It emits a patterned light (usually infra-red light) and measures the disparity of the


72
168
134
unit in mm
x5
x4
y5
z4
z5
y4
x3
70
y3
z3
x6
z6
y6
m2
270
Frame $\mathscr{F}_4$ & $\mathscr{F}_6$ is shifted backward by the DH algorithm.
x2
z2
z1
125
x1
y1
z0
165
x0
y0

(a) (b) (c)

Fig. 2: (a) ABB IRB 120 point cloud model - the (joint) frames are set-up by DH convention. Each link is implemented as an independent model but coupled to its neighbor over the respective DH transformation; (b) ABB IRB 120 kinematic structure; (c) Robot depth image captured by two Intel® RealSense R200 in a distance of 1.5 m.

captured pattern in comparison to the original one to get the depth information by triangulation. Third, Time-of-Flight (ToF) cameras [17] where the depth information is measured over the elapsed time of pixel-wise emitted modulated light signals reflected by the detectable object.

RGB-D cameras provide point clouds (with position information in $\mathbb{R}^3$) generated from depth data. Thus, it is not necessary to seek for interest points for orientation estimation since the detected objects are already available as 2.5D objects in the workspace. Similar to an image-based approach a 3D model of an object can be matched to the point cloud via the Iterative Closest Point (ICP) algorithm [18–20] to find its alignment. It minimizes the distance between two point clouds with the requirement that the two point clouds are roughly close to each other (the initial guess), until they are aligned. The ICP algorithm consists of the following phases:

- **Selection** of point pairs,
- **Matching** of these point pairs,
- **Rejection** of point pairs due to individual consideration,
- **Error metric** assignment,
- **Minimizing** the error metric.

With the ICP algorithm, an alignment can be achieved within a few iterations.

Now, the idea is, instead of matching the whole robot as a rigid body, to split the robot into its links and match them separately (cf. Figure 1) in an eye-to-hand composition, such that the orientation of its joints can be estimated. In this case the use of markers can be omitted since the joint orientation can be calculated from the alignment of the links to each other, which makes this approach a versatile applicable method for industrial applications. The knowledge of the robot's kinematic chain gives the possibility of robot pose variation by well-defined joint orientations as well as the variation of the joint orientation during motion to correct the trajectory in case of work-flow disturbance. The goal of this approach is a visual servoing concept by depth sensing with a potential to collision protection and avoidance in a collaborative applicable manner.

This paper is organized as follows. In Section II, the applied method is described. The implementation of the robot's link point cloud models is described in Section III. The description of the setup and the camera alignment is described in Section IV. In Section V, the presented work is summarized and Section VI concludes the paper.

II. METHODS & APPROACH

The goal of the presented approach is to track a manipulator with six Degrees-of-Freedom (DoF) by two RGB-D imaging systems for joint position perception and visual servoing. For the measurement of the robot's joint alignment, the cameras are placed in an eye-to-hand composition. This allows to capture the whole manipulator from a wider view and avoid occlusions. The depth sensing technology with the highest accuracy for positioning and object matching is derived by comparing two different camera technologies. Therefore, a structured light camera and a ToF camera is applied and tested. Before the pose of the robot can be estimated, the position of both cameras have to be extrinsically calibrated, to get a perfect aligned point cloud from both cameras. The camera calibration is carried out as a transformation of the camera coordinate system by its physical position relative to the robot's base coordinate system.

For a matching process of point clouds by an appropriate approximation method like ICP to receive the robot's joint positions as mentioned in Section I, the models of its links, generated from Computer-Aided Design (CAD) files, have to be prepared. This is done by aligning the link models in the CAD files in their initial position as shown in Figure 2a and

TABLE I: Denavit-Hartenberg parameter

Name	Symbol	Description
joint angle	θ_i	angle between x_{i-1} and x_i about z_{i-1}
link offset	d_i	distance between the origin of frame $\mathscr{F}_{i-1}$ and $\mathscr{F}_i$ along z_{i-1}
link length	a_i	offset between frame $\mathscr{F}_{i-1}$ and $\mathscr{F}_i$ along x_i
link twist	α_i	angle between z_{i-1} and z_i about x_i

TABLE II: Robot link properties

Parameter	Class
Name	std::string
Point cloud	pcl::PointCloud<PointXYZRGBA>*
Color	pcl::visualization::PointCloudColor-HandlerCustom<PointXYZRGBA>*
DH-parameter	std::vector<double>
DH-transformation matrix	Eigen::Matrix4f
Joint angle	std::double

subsequently the generation of point cloud representation. The dependence of each links pose to each other in the model will be set-up by applying the Denavit-Hartenberg (DH) convention to achieve the kinematic chain as shown in Figure 2b. The DH convention describes the transformation between two frames of a manipulator by a homogeneous transformation matrix ${}^{i-1}\boldsymbol{T}_i \in \mathbb{R}^{4\times4}$ with four parameters by placing the joint coordinate frames in a predefined way. These transformations are represented by four basic transformations between the joints as a chain of two rotations and two translations

$$ {}^{i-1}\boldsymbol{T}_i = \mathrm{Rot}_{z_{i-1},\theta_i}\mathrm{Trans}_{z_{i-1},d_i}\mathrm{Trans}_{x_i,a_i}\mathrm{Rot}_{x_i,\alpha_i} \ , \tag{1} $$

with the DH parameters listed in Table I.
This convention will simplify the calculation effort for matching via the ICP algorithm to only one DoF per joint and keep the links dependent from each other. The deviation from the robot's point cloud to the model is used for the calculation of the joint velocities to align both point clouds again. The whole implementation is realized with the free Point Cloud Library (PCL) [21], which includes numerous algorithms for handling of n-dimensional point clouds and three-dimensional geometries, in the framework of the Robot Operating System (ROS) [22]. ROS is a collection of libraries, tools and conventions for writing robot operating software.

III. MODEL IMPLEMENTATION

In an initial step point clouds from the CAD models of the robot's links have to be generated. It is important that, before the point clouds can be generated, the alignment of the CAD modeled links are prepared correctly as mentioned in Section II. First, they have to be aligned in their initial direction (cf. Figure 2b), second, their coordinate system must be set to the center of their rotation axis, and third, the link coordinate systems have to be translated such that they match with the DH convention as it is done for link four and six (translation in x direction) as shown in Figure 2b. The point clouds are generated by the tool *pcl_mesh2pcd* (based on *take views and fuse them together*) from the PCL to achieve an envelope point cloud of the CAD models.

Every link is implemented as an own object with the properties summarized in Table II, with the first four parameters as constants and the transformation matrix and joint angle as variables. The robot's links will not separate from each other during the ICP algorithm performs the matching, since they are coupled by the transformation of DH with the parameters of Table III and the dependencies given by

$$ {}^{0}\boldsymbol{T}_n = \prod_{i=1}^{n} {}^{i-1}\boldsymbol{T}_i \ , \tag{2} $$

$$ \boldsymbol{T}_{n,\alpha} = {}^{0}\boldsymbol{T}_n \boldsymbol{T}_\alpha {}^{n}\boldsymbol{T}_0 \ , \tag{3} $$

$$ {}^{n-1}\boldsymbol{T}_n = {}^{n-1}\boldsymbol{T}_0 \boldsymbol{T}_\alpha {}^{0}\boldsymbol{T}_n \ . \tag{4} $$

In Equation (2), ${}^{0}\boldsymbol{T}_n \in \mathbb{R}^{4\times4}$ is the transformation of joint n between the base coordinate system and the coordinate system of joint n as the product of the DH transformations. By the use of the short notation $\mathrm{c}(\cdot) = \cos(\cdot)$ and $\mathrm{s}(\cdot) = \sin(\cdot)$, the DH transformation matrix ${}^{i-1}\boldsymbol{T}_i$ from Equation (1) can be written as

$$ \begin{aligned} {}^{i-1}\boldsymbol{T}_i &= \begin{bmatrix} {}^{i-1}\boldsymbol{R}_i & {}^{i-1}\boldsymbol{p}_i \\ \boldsymbol{0}^\mathrm{T} & 1 \end{bmatrix} \\ &= \begin{bmatrix} \mathrm{c}_{\theta_i} & -\mathrm{s}_{\theta_i}\mathrm{c}_{\alpha_i} & \mathrm{s}_{\theta_i}\mathrm{s}_{\alpha_i} & a_i\mathrm{c}_{\theta_i} \\ \mathrm{s}_{\theta_i} & \mathrm{c}_{\theta_i}\mathrm{c}_{\alpha_i} & -\mathrm{c}_{\theta_i}\mathrm{s}_{\alpha_i} & a_i\mathrm{s}_{\theta_i} \\ 0 & \mathrm{s}_{\alpha_i} & \mathrm{c}_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \ , \end{aligned} \tag{5} $$

with ${}^{i-1}\boldsymbol{R}_i \in \mathbb{R}^{3\times3}$ the rotation between the frame $\mathscr{F}_{i-1}$ and $\mathscr{F}_i$, the translation ${}^{i-1}\boldsymbol{p}_i \in \mathbb{R}^{3\times1}$ from the origin $\mathscr{O}_{i-1}$ to $\mathscr{O}_i$ and the vector of zeros $\boldsymbol{0} \in \mathbb{R}^{3\times1}$. $\boldsymbol{T}_{n,\alpha} \in \mathbb{R}^{4\times4}$ in Equation (3) is the transformation of joint n in the base coordinate system by the transformation matrix

$$ \boldsymbol{T}_\alpha = \begin{bmatrix} \boldsymbol{R}_z(\alpha) & \boldsymbol{0} \\ \boldsymbol{0}^\mathrm{T} & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \ , \tag{6} $$

with the rotation matrix $\boldsymbol{R}_z(\alpha) \in \mathbb{R}^{3\times3}$ along the joint rotation axis which is obtained from the Euler angles by the ICP

TABLE III: Denavit-Hartenberg parameters of the industrial robot ABB IRB 120

JointNr.	θ_i [°]	d_i [mm]	a_i [mm]	α_i [°]
1	q_1	165	0	0
2	q_2	125	0	$-\pi/2$
3	$q_3 - \pi/2$	0	270	0
4	q_4	0	70	$-\pi/2$
5	q_5	302	0	$\pi/2$
6	q_6	0	0	$-\pi/2$

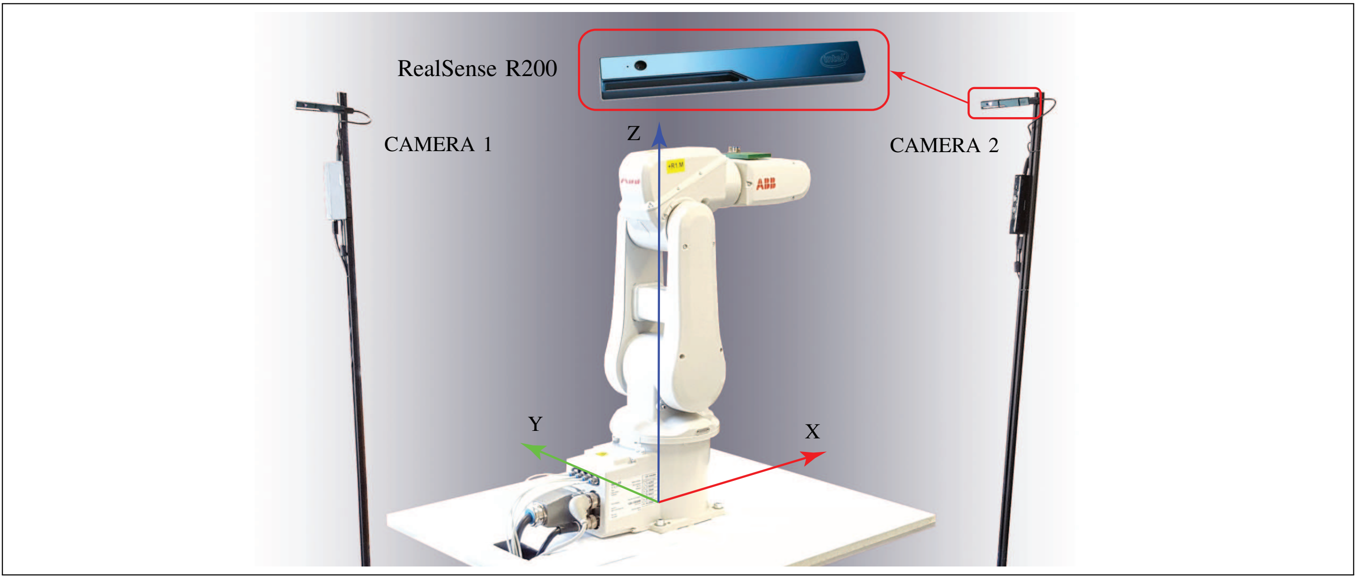


Fig. 3: Experimental setup with two Intel® RealSense R200 structured light cameras in 90° alignment to an ABB IRB 120.

algorithm. $\boldsymbol{R}_z(\alpha)$ performs a roll, pitch, or yaw rotation about the angle α according to the joint rotation axis. The new DH-transformation matrix ${}^{n-1}\boldsymbol{T}_n \in \mathbb{R}^{4\times 4}$ of Equation (4) will be saved after the rotations have been performed. By iteration of these equations every rotation $\boldsymbol{R}_z(\alpha)$ of a joint n will be passed to the following joints. This guarantees that every joint keeps coupled to each other.

IV. SETUP & CAMERA ALIGNMENT

Two identical structured light cameras (Intel® RealSense R200) are used in the setup (cf. Figure 3) to avoid occlusions and to get a denser point cloud representation of the robot as shown in Figure 2c. The cameras are positioned in 90° to each other. This angle has been chosen since the influence of the illumination disturbance by the projected structured lights is minimized. Each camera is placed 65 cm above the robot with a pitch angle of 30° down to have a wider view. The extrinsic camera calibration will be performed through a plane calibration. Therefore, the table where the robot is placed on has been detected by the outliers' detection method Random Sample Consensus (RANSAC) to receive the model coefficients of the plane $\mathscr{A}_{xy}$. With the model coefficients, the dihedral angle between the plane normal and camera image normal can be derived by the equation

$$cos(\varphi) = \frac{\vec{n}_1 \cdot \vec{n}_2}{|\vec{n}_1| \cdot |\vec{n}_2|} \tag{7}$$

where $\vec{n}_1 = (a_1, b_1, c_1)$ is the normal vector of the plane $\mathscr{A}_{xy}$ in z direction and $\vec{n}_2 = (a_2, b_2, c_2)$, the normal vector of the camera image plane $\mathscr{A}_{yz}$ along the x direction with the plane coefficients a_i, b_i, c_i for $i = 1, 2$. The cameras are aligned by the rotation with φ from Equation (7) (plus the camera pitch angle) and the known translation from the robot's base.

V. EVALUATION & RESULTS

The test system, which is used to evaluate the proposed approach, consists of a personal computer with an Intel® Core™ i5-3470 @ 3.20 GHz, 4096 MB RAM, and a GeForce GT 630 with the operation system Linux Ubuntu 16.04 @ 64 bit.

TABLE IV: The parameters used for the Iterative-Closest-Point algorithm

Max. Correspondence Distance	Max. Iterations	Transformation Epsilon	Euclidean Fitness Epsilon
0.003 m	100	1e-8 m	5e-4 m

So far, the structured light cameras and the robot motion communication are implemented successfully in ROS. The cameras and the robot are launched as ROS nodes such that they can communicate with each other. The Point cloud models of the robot's links are generated from CAD files and coupled together via the DH convention such that they depend on each other and that a rotation of joint one, for instance, has an effect to the other joints (cf. Equations (2) to (4)). A visualization is implemented to visualize the model together with the captured depth image as shown in Figure 1. The joint positions and alignments from the implemented model are observable and controllable. Since the ICP algorithm needs an initial guess where it should start the matching, an initial robot position for program start has been chosen as shown in Figure 2a, otherwise a correct estimation of the position would be hardly possible. In the first experiment the built-in ICP algorithm from the PCL has been tested with the parameters from Table IV and structured light cameras with moderate results. While for the initial pose (start pose) reasonably accurate joint angles with $\pm 0.5°$ have been measured, the deviation increased up to $\pm 5°$ during motion. These evaluation results were obtained for slow motion tasks ($\leq 1\,°/\mathrm{s}$). For faster movements the ICP algorithm is not able to finish the required number of iterations on the test system and the point cloud model can not be matched. The low accuracy of the ICP algorithm in the experiments for low speeds may occur to the very bumpy surface images from the cameras (Figure 1), which makes it difficult to calculate an accurate match. A smoothing of the robot's point cloud by the moving least squares method from the PCL also does not significantly improve the results, since the outliers' in the robot's point cloud surface are too large (cf. Figure 2c) to achieve good results.

VI. CONCLUSION & OUTLOOK

A robot point cloud model generated from CAD data for each robot link have been adopted and linked via the DH convention. A linked motion algorithm is integrated so that each link depends from each other. The first tests with structured light cameras and the ICP algorithm from the PCL showed moderate results. For the next tests with structured light cameras, the results should be improved by the implementation of a Levenberg-Marquardt Optimizer [23, 24] for an optimized registration. The change of the camera system to ToF cameras will also bring better results with the general ICP algorithm. So far the operation area is limited by only two cameras, because the robot's tool center point is not detectable overall by reason of occlusions in negative y-direction. A remedy would be to place a third camera right from the robot. This is feasible with a ToF camera but challenging with a structured light camera due to illumination disturbance from the counterpart. An alignment of 60 degrees for three structured light cameras would be better, since all the three cameras would receive the same disturbance which is less than if two of three fully receive it. A faster and more general model implementation would bring the implementation of an automatic model generation from COLLAborative Design Activity (COLLADA) [25] data which can be generated easily by CAD programs. With the COLLADA data (version 1.5.0) not only the geometry parameter would be loaded, the mechanical parameter as mass, inertia and center of mass could be loaded too, which is interesting for the robot dynamic. This would remove the model preparation as mentioned in Section III for a more user-friendly application.

REFERENCES

[1] J. O'Neill, J. Lu, R. Dockter, and T. Kowalewski, "Practical, stretchable smart skin sensors for contact-aware robots in safe and collaborative interactions", in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2015, pp. 624–629.

[2] C. Liu, Y. Huang, P. Liu, Y. Zhang, H. Yuan, L. Li, and Y. Ge, "A flexible tension-pressure tactile sensitive sensor array for the robot skin", in *Robotics and Biomimetics (ROBIO), 2014 IEEE International Conference on*, IEEE, 2014, pp. 2691–2696.

[3] A. De Luca and R. Mattone, "Sensorless robot collision detection and hybrid force/motion control", in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, IEEE, 2005, pp. 999–1004.

[4] K. Kosuge and T. Matsumoto, "Collision detection of manipulator based on adaptive control law", in *Proc. IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics*, 2001, pp. 117–122.

[5] C. Morato, K. N. Kaipa, B. Zhao, and S. K. Gupta, "Toward safe human robot collaboration by using multiple kinects based real-time human tracking", *Journal of Computing and Information Science in Engineering*, vol. 14, no. 1, p. 011 006, 2014.

[6] B. Schmidt and L. Wang, "Depth camera based collision avoidance via active robot control", *Journal of Manufacturing Systems*, vol. 33, no. 4, pp. 711–718, 2014.

[7] A. Muis and K. Ohnishi, "Eye-to-hand approach on eye-in-hand configuration within real-time visual servoing", *IEEE/ASME Transactions on Mechatronics*, vol. 10, no. 4, pp. 404–410, 2005.

[8] F. Janabi-Sharifi, L. Deng, and W. J. Wilson, "Comparison of basic visual servoing methods", *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 5, pp. 967–983, 2011.

[9] B. Thuilot, P. Martinet, L. Cordesses, and J. Gallice, "Position based visual servoing: Keeping the object in the field of vision", in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, IEEE, vol. 2, 2002, pp. 1624–1629.

[10] T. Koenig, Y. Dong, and G. N. DeSouza, "Image-based visual servoing of a real robot using a quaternion formulation", in *Robotics, Automation and Mechatronics, 2008 IEEE Conference on*, IEEE, 2008, pp. 216–221.

[11] E. Marchand and F. Chaumette, "Visual servoing through mirror reflection", in *IEEE Int. Conf. on Robotics and Automation, ICRA'17*, 2017.

[12] D. Tsai, D. G. Dansereau, T. Peynot, and P. Corke, "Image-based visual servoing with light field cameras", *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 912–919, 2017.

[13] N. Shahriari, S. Fantasia, F. Flacco, and G. Oriolo, "Robotic visual servoing of moving targets", in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, IEEE, 2013, pp. 77–82.

[14] H. Liu, S. Huang, N. Gao, and Z. Zhang, "Binocular stereo vision system based on phase matching", in *SPIE/COS Photonics Asia*, International Society for Optics and Photonics, 2016, 100230S–100230S.

[15] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review", *IEEE Transactions on Cybernetics*, vol. 43, no. 5, pp. 1318–1334, 2013.

[16] S. K. Nayar and M. Gupta, "Diffuse structured light", in *Computational Photography (ICCP), 2012 IEEE International Conference on*, IEEE, 2012, pp. 1–11.

[17] S. Foix, G. Alenya, and C. Torras, "Lock-in time-of-flight (ToF) cameras: A survey", *IEEE Sensors Journal*, vol. 11, no. 9, pp. 1917–1926, 2011.

[18] P. J. Besl and N. D. McKay, “Method for registration of 3-D shapes”, in *Robotics-DL tentative*, International Society for Optics and Photonics, 1992, pp. 586–606.

[19] S. Rusinkiewicz and M. Levoy, “Efficient variants of the ICP algorithm”, in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, IEEE, 2001, pp. 145–152.

[20] A. Aldoma, M. Vincze, N. Blodow, D. Gossow, S. Gedikli, R. B. Rusu, and G. Bradski, “CAD-model recognition and 6DOF pose estimation using 3D cues”, in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, IEEE, 2011, pp. 585–592.

[21] R. B. Rusu and S. Cousins, “3D is here: Point cloud library (PCL)”, in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[22] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “ROS: an open-source robot operating system”, in *ICRA workshop on open source software*, Kobe, vol. 3, 2009, p. 5.

[23] S. Roweis, “Levenberg-Marquardt Optimization”, *Notes, University Of Toronto*, 1996.

[24] Y. Wu, W. Wang, K. Lu, Y. Wei, and Z. Chen, “A new method for registration of 3D point sets with low overlapping ratios”, *Procedia CIRP*, vol. 27, pp. 202–206, 2015.

[25] M. Barnes and E. L. Finch, “COLLADA - digital asset schema release 1.5.0”, *Sony Computer Entertainment Inc. Atazadeh, Sam Amirebrahimi, Alireza Jamshidi (7048) 3D-Cadastre, a Multifaceted Challenge*, 2008.